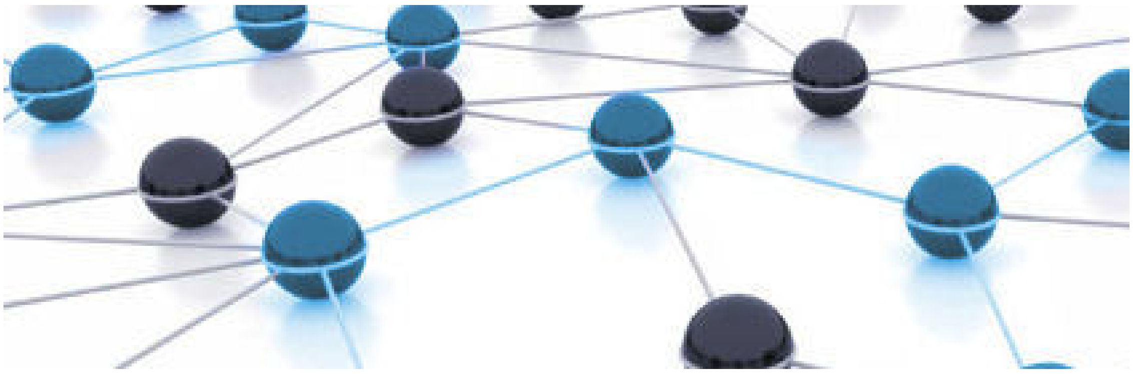
TÉCNICO LISBOA



Autonomous system for track athletics guidance for the visual impaired

Bernardo de Matos Patrocínio dos Santos

Thesis to obtain the Master of Science Degree in

Telecommunications and Informatics Engineering

Supervisors: Prof. Ricardo Jorge Fernandes Chaves
Prof. José Eduardo Charters Ribeiro da Cunha Sanguino

Examination Committee

Chairperson: Prof. Paulo Jorge Pires Ferreira
Supervisor: Prof. Ricardo Jorge Fernandes Chaves
Member of the Committee: Prof. Nuno Miguel Carvalho dos Santos

October 2014

Acknowledgments

My academic journey was not easy to endure, but at the end I made to the end. The same goes regarding this work that despite not being easy, it was a very interesting and exciting subject to research and develop for. Despite my personal achievements during the process of this journey, this also would not be possible without the help and support of the following individuals:

To my mother **Luísa** and **António**, for being there for me always by my side, supporting me through hard times and cheering through the good times. The same goes to my father **Pedro** who always has been there for me and made sure that nothing was missing in order to achieve my success. A special thank you for my aunt **Clara** and uncle **António** for also supporting me through my journey and for always encouraging me to continue on to pursue my dreams.

To my supervisors, **Prof. Ricardo Chaves** and **Prof. José Sanguino** a special thank you for believing in my work and potential and for making sure that I wouldn't slack off in the work that this thesis demanded. It wasn't easy, but we made it possible. A special thank you also to **Prof. Rui Santos Cruz** for also encouraging to pursue my career choices and for being always available for any academic conundrum that I would stumble upon.

To **Soraia Meneses Alarcão**, my battle companion in this academic war, my confident and most of all my best friend, we have been through a lot, but we always believed in each other's success at the end. I am so proud of you and I'm thankful for having you as a friend.

Also to my friends **Ana Pais**, **Andreia Ferrão**, **Dina Pires**, **Dinamene Barreira**, **Inês Fernandes**, **Inês Castelo**, **João Murtinheira**, **Joana Camacho**, **Joana Condeço**, **Luis Carvalho**, **Margarida Alberto**, **Miguel Coelho**, **Nuno Fernandes**, **Ricardo Carvalho**, **Rui Fabiano**, **Tiago Esteves** and **Vânia Mendonça**, a special super thank you for your constant support, you guys are the best!

To each and every one of you – Thank you.

Abstract

Currently, visually impaired people still suffer from the lack of adequate pedestrian guidance systems suited to their needs and without which they are not able to have a good quality of life, something that must be addressed urgently. Visually impaired athletes, more specifically, are not able to train and practise their runs due to increasing costs regarding guide athletes, infrastructures and needed equipment.

In this work, available positioning/navigation techniques and guidance systems were evaluated regarding a set of requirements demanded for the specified target audience. Although the solutions mentioned have some key aspects developed, there is no system that gathers all the required functionalities.

It is with that motive that it is proposed the **TSAG** system, a navigation guidance system for visually impaired athletes, that provides vibratory and audio feedback to an athlete during his competition, while using Pedestrian Dead Reckoning (PDR) techniques to track its position and correct it if it deviates from the course.

Keywords

accelerometer, Android, Dead Reckoning (DR), gait, Global Positioning System (GPS), gyroscope, kinetic

Resumo

Actualmente, pessoas que sofram de algum tipo de insuficiência visual não possuem meios para poder navegar de forma mais segura e livre, limitando a sua qualidade de vida, e é algo que deve ser tratado com celeridade. No caso particular do atletas paraolímpicos invisuais, os mesmos não são capazes de treinar/praticar devido aos custos elevados que os equipamentos e instalações próprias para o efeito têm.

Neste trabalho, são analisadas técnicas e algoritmos de posicionamento e navegação pedestre considerando um conjunto de requisitos que vão ao encontro das necessidades do público-alvo mencionado anteriormente. Embora existam sistemas capazes de resolver alguns assuntos relacionados com a navegação para pessoas invisuais, é de notar a inexistência de uma solução composta com todos os módulos necessários para a total liberdade do atleta.

É por este motivo que é proposto o sistema **TSAG**, um sistema de guia e navegação para atletas invisuais, fornecendo feedback auditório e vibratório ao atleta durante a sua prova, enquanto a algoritmia PDR embutida no sistema rastreia a sua posição e corrige-a caso o mesmo se desvie do percurso correcto.

Palavras Chave

acelerómetro, Android, DR, passada, GPS, giroscópio, cinética

Contents

List of Figures

List of Tables

List of Algorithms

Listings

Acronyms

API	Application Programming Interface
GPS	Global Positioning System
GIS	Geographic Information System
DGPS	Differential Global Positioning System
RTK	Real Time Kinematic
PPP	Precise Point Positioning
DR	Dead Reckoning
PDR	Pedestrian Dead Reckoning
ETA	Electronic Travel Aids
EOA	Electronic Orientation Aids
MEMS	Micro-Electro-Mechanical-Systems
MOS	Mobile Operating System
NFC	Near Field Communication
TTS	Text-To-Speech

1 Introduction

Contents

It is estimated that, in 2013, over 4%[1] of the world's population (285 million) had visual impairments and one of the main consequences of such disability is the loss of autonomy in terms of navigation, something that these people have to deal with every day. In order to perform certain activities that require to go from point A to point B, people with visual impairments can only rely on their walking sticks/canes or, in certain cases, specialised *seeing eye dogs*.

However, these current solutions have their limitations since, for instance, the walking sticks/canes give a very narrow range (navigation wise) for the visually impaired person, who also has to learn how to correctly collect, process and react accordingly regarding the information that is provided from the equipment. The same goes for the *seeing eye dogs*, since they require a lot of time and money to be trained for these specific guidance tasks and added responsibilities.

This is also verified when it comes to physical activity, since they do not have the same opportunities to participate in regular sports and do not attain the social and physical benefits since their main barriers are time, the lack of adapted facilities and equipment, human resources, and monetary cost. This is very noticeable when we consider visually impaired running athletes, who on top of all the mentioned above barriers, also need a runner guide, which will lead to more costs and the athlete is limited to the guide's performance and stamina.

However, the increasing evolution of technology (especially regarding the **navigation**, **mobility** and **accessibility** areas) has come to a point where it can be possible to consider the development of a portable navigation system that can address this shortcoming.

1.1 Motivation

Current navigation systems and sensor technologies allow us to overcome some of these difficulties. With this motivation, the purpose and goal of this work is to address and develop a portable, autonomous and precise guidance system for visually impaired athletes to allow them to autonomously practice and compete in various physical activities, mainly running sports.

Nowadays, only a small minority of developed systems that were addressing this issue are still being used, mainly due to the lack of precision, since general GPS devices available for a civilian use can only provide a precision of 20 meters (radius), inadequate interaction with the user, the (high) cost of the available systems and the deficiency of usage, since some of the available technologies are old and

[1]Source: World Health Organisation (WHO): http://www.who.int/mediacentre/factsheets/fs282/en/ - Last Visited: 10/2014

do not present some important features that are needed in order to tackle this situation. Although there are some promising techniques (mostly positioning algorithms), and as stated in 1, with the increasing evolution of technology, we are able to develop a new system that can take advantage of all the available Precise Point Positioning (PPP) techniques (but also refurbish them a bit to best suit the user's needs) and ally them to a low-cost but efficient device.

1.2 Requirements

An automated guidance system will allow visually impaired athletes to train in an autonomous fashion whenever they need or whenever the guide runner cannot be present, allowing a significant increase in the availability of this so common sport. In order to do so, it is of key importance to adequately define the specifications of the system, namely: i) develop a **low-cost** all-in-one solution, ii) develop a solution for a **portable** device, iii) develop a solution with adequate **positioning precision** algorithms iv) develop an all-in-one solution which brings an overall **ease of usability** and v) a device where it is possible to **develop** applications for its software/firmware. Nowadays, smartphones already include a GPS receiver, sensors such as **accelerometers** and **gyroscopes**, wireless connections and also audio/vibratory interfaces.

Due to the conditions of the running track, the competition and the athlete themselves, it is important to assure the following while developing the system:

- the **improvement** of the **precision** in the positioning estimates of low-cost GPS receivers with the usage of techniques such as Real Time Kinematic (RTK) or PPP;
- the **improvement** of the **guidance position** in devices with simple GPS data by developing DR techniques that explore specific motion characteristics of the athletes and the racing track geometry;
- the **development** of a system that will be engaged and able to interact with several devices, such as beacon stations to transmit data to use in RTK and PPP techniques, but also with several mobile nodes (the athletes);
- the **definition** and **implementation** of adequate interfaces with the system;

1.3 Goals

The main goal of this work is to develop an autonomous guidance system for visually impaired athletes with low-cost and portable embedded sensors, such as *smartphones*, allowing visually impaired athletes to run without the guide runner. The system must be autonomous, and with enough precision to guide the athlete within the track limits. At the same time, it must be portable, economically viable and accessible, and with an appealing interface that is also easy to use, comprehend and interact with.

The system will be designed considering outdoor running tracks and the path to be followed by the athlete must be predefined, well known and free of obstacles but it must provide a fast response since the athlete will be at a fast pace (up to 10 m/s). With such a guidance system, it is possible to facilitate physical activities, in particular running sports, to be available to more people with visual impairments.

It is necessary to use some of the available high-accuracy positioning algorithms mentioned above (section 1.2), to evaluate the athlete's status during the race and also to be able to gather information regarding the athlete's gait. It is also necessary to develop a **calibration algorithm** since each athlete has its own conditions (age, height, pace and weight).

The system must also provide feedback at all times due to the athlete's disability, so it can inform the athlete of its current position in the track (during the race) and it must provide useful alerts if the athlete happens to deviate for the course. All of this must happen while guiding the athlete during the trial (e.g. if the athlete gets sidetracked during the race, the vibration device of the equipment will inform it of such event and the system will provide directions (speaking aids) so it can get back to the original course). The positioning algorithms developed for the system can be allied with the equipment's GPS receiver in order to provide the best measurement of its current position.

1.4 Contributions

Regarding our requirements and goals defined in 1, we have made the following contributions:

- a **tracking** component that has obtained accuracy levels over 95% on step estimation and over 75% on distance estimation, which establish a slight **improvement** over the current algorithms and systems available for a **portable** environment;
- a **positioning** component that is able to provide positioning coordinates at a maximum error of $10m$ (surrounded by obstacles that obstruct the capture of the GPS signal), and obtaining also accuracy levels (for the distance between the first and last coordinate set obtained) over 82%, making the usage of the GPS (location provider and receiver) stable, fair and fitted for this system;
- the creation of a **feedback provider**, allied with the Text-To-Speech (TTS) module and vibratory patterns that allow to guide the athlete throughout the entire race without any auxiliary aid;

- Finally, the creation of a **low-cost** all-in-one solution, in which all the components and features mentioned above were developed for a **portable** device with built-in sensors and receivers.

1.5 Document Structure

This thesis is organized as follows: In chapter 2, all state of the art solutions able to tackle and solve some parts of the problem are presented as well a review of the available algorithms that were developed. In chapter 3, the **TSAG** system, the proposed solution is presented, with a description of its main features (design and functionality) as well as other considerations that need to be taken into account. In order to solve the problem presented in chapter 1, in chapter 4, the proposed solution is described regarding its implementation, taking into consideration all the algorithms and libraries used and developed for the system. In order to validate the work developed, in chapter 5, is presented the tests made to the system considering some scenarios. Finally, in chapter 6, final notes and considerations regarding the system are made, considering also possible future work that can be developed.

2
Related Work

Contents

In order to be able to develop a solution that will be suitable for our purpose, it is important to pinpoint the criteria needed while taking in consideration the requirements established in 1.2. It is necessary to consider such a **device** that is reliable, resistant to constant motion while practising sports (e.g. running) and it must also be able to provide such accurate and precise positioning information for the targeted audience. In such device, a **set of built-in sensors** will be needed in order to provide the most precise and accurate positioning results due to their output data allied with the existing PDR techniques/algorithms.

Hence it is necessary to consider and evaluate the solutions (academic and commercial) available as well as other relevant related work and/or technology in order to understand the strengths and weaknesses of each available solution, but knowing *a priori* that most of them are not prepared for the fast pace motion that running sports require. This will lead to the development and implementation of a solution that matches the desired criteria.

2.1 Supporting Platforms/Devices and Existing Sensors

Guidance systems for visually impaired people can be divided into two main categories [1]: **micronavigation**, which are designed to focus on the detection of obstacles in a certain path and **macronavigation**, whose purpose is to find a path in a large environment.

Electronic Travel Aids (ETA), used for micronavigation, tries to identify objects and in some cases, the direction of these objects in the near field of the guided person. These existing systems are based on the emission of acoustic or light signals using ultrasounds, lasers, or infrared light, typically mounted on the head, neck, belt, or on the walking stick/cane. This detection is possible in both indoor and outdoor environments, with ranges from 1 up to 15 meters and with angles from 30 to 45 degrees. Micronavigation is a complementary system usually used along with a seeing eye dog or a walking stick/cane. The output interfaces uses audio tones or voice and vibration *stimuli*. The input is performed by switches used to control the perception distance, volume, and other minor features. [1], [2]

Electronic Orientation Aids (EOA), used for macronavigation, aids the user in finding his path in known regions but also in unknown regions never crossed before. The key component in the existing systems is the use of GPS in order to determine the exact location of the user. Several systems have been proposed and developed using GPS or Geographic Information System (GIS) to guide the user [3], [4], [5]. The user is informed of the direction by speech, sounds or vibration actuators. The major problems of these systems are the obtained adequate positioning precisions and the absence of maps in some regions or maps oriented for pedestrian guidance.

In order to satisfy the requirements established in 1.2, it is recommended to choose a ***smartphone***, a mobile phone with advanced computing capability and connectivity. Today's smartphones have various features such as portable media players, low to mid-end compact digital cameras, GPS navigation units and built-in sensors such as the **gyroscope** (herein considered for the DR approach) and the **accelerometer**, high-resolution touchscreens and web browsers that display standard web pages as well as mobile-optimized sites. Regarding the connectivity, high-speed data accesses are provided by Wi-Fi, mobile broadband, Near Field Communication (NFC), and also Bluetooth.

2.2 Mobile Operating Systems

As mentioned in 2.1, the solution will be developed and implemented taking into account a smartphone. A smartphone has a built in Mobile Operating System (MOS), which means that it it able to combine the features of a personal computer operating system with other features such as a touchscreen, cellular network, Bluetooth, WiFi, GPS mobile navigation, camera (photo and video), speech recognition and text-to-speech functions, voice recorder, music player, NFC, and Infrared interfaces. For the proposed solution, it was important to consider the most versatile and commonly available MOS nowadays, that we will briefly describe in the following subsections (or paragraphs):

2.2.1 Android

Android is a MOS based on the Linux kernel and designed primarily for touchscreen mobile devices such as smartphones or even tablets. Its user interface is based on direct manipulation, using touch inputs that loosely correspond to real-world actions like ”swiping”, ”tapping” or ”pinching” to manipulate on-screen application features. Allied with the smartphone's internal hardware, especially its sensors such as accelerometers, gyroscopes and proximity sensors, it is possible to be used by some applications to respond to additional user actions, for example, adjusting the screen from portrait to landscape, depending on how the device is oriented. It is *open-source* and it was released under the Apache License. This open-source code and permissive licensing allows the software to be freely modified and distributed by device manufacturers, wireless carriers and enthusiast developers. Android has a large community of developers writing applications that extend the functionality of devices, written primarily in the Java programming language.[1]

[1]Source: `http://developer.android.com/about/index.html` - Last Visited: 10/2014

2.2.2 iOS

iOS is a MOS developed and distributed by Apple Inc. Its user interface is based on the concept of direct manipulation using multi-touch gestures. Like the Android MOS, the interaction with the OS includes gestures such as swipe, tap, pinch, and reverse pinch, all of which have specific definitions within the context of the iOS operating system and its multi-touch interface. Accelerometers and other sensors are used by some applications to respond to shaking the device or rotating it in three dimensions. Though it is massively popular, it can only be used on devices manufactured by Apple and it is not open-source.[2]

2.2.3 Windows Phone

Windows Phone (abbreviated as WP) is a series of proprietary smartphone operating systems developed by Microsoft. It features a user interface based on tiles that link to applications, features, functions and individual items (such as contacts, web pages, applications or media items). It takes advantage of multi-touch technology, multi-core processors, NFC (which can primarily be used to share content and perform payments) and also support for removable storage.[3]

2.3 Existing Sensors

Smartphones have become very popular and indispensable carry-on devices for people in recent years, and they are embedded with various sensors which could be used for many interesting applications. Sensors in smartphones can provide unlimited possibilities for applications to help and change the life of people. With that, developing an application that interacts with various types of sensors is easier to be accepted instead of creating and developing for specific equipments [6], given the improvement of Micro-Electro-Mechanical-Systems (MEMS) and the fact that the presence of sensors (with an improved component integration to increase performance with reduced costs) in mobile phones has risen exponentially in order to provide new features/services to end-users. [7]

The typical types of sensors built-in in today's smartphones are (as illustrated in Figure 2.1):

- **Motion sensors**: With the usage of accelerometers, gravity sensors and gyroscopes, it is possible to measure acceleration forces and rotational forces along three axes.
- **Environmental sensors**: These sensors (such as barometers and thermometers) measure various environmental parameters such as ambient air temperature and pressure or even, in some

[2]Source: http://www.apple.com/ios/what-is/ / Last Visited: 10/2014
[3]Source: http://www.windowsphone.com/en-us/features/all - Last Visited: 10/2014

cases, humidity.

- **Position sensors**: These sensors measure the physical position of a device. This category includes orientation sensors and magnetometers.

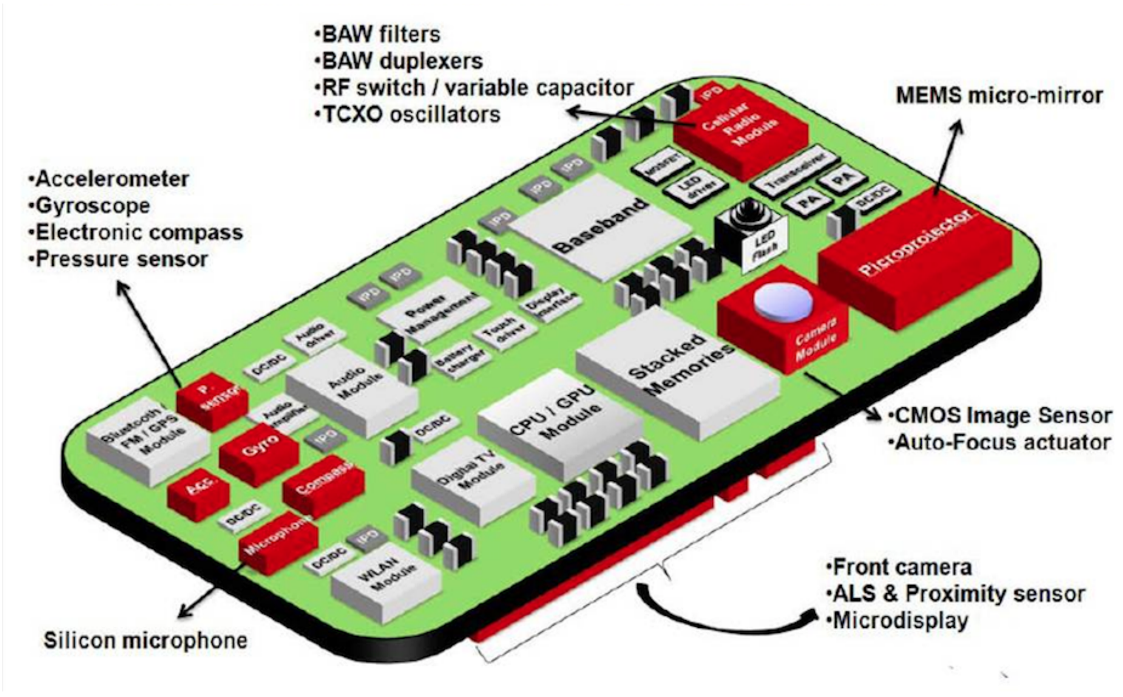


Figure 2.1: Example (schematics) of a smartphone with built-in sensors [8]

2.3.1 Accelerometer

An accelerometer is a sensor that is able to detect the forces the sensor is subjected to. It is also able to determine the orientation of a device towards the ground by measuring the gravitational force. As an example, a value of 1 indicates that the device is experiencing $1G$ of force exerting on it over the YY axis. Nowadays, smartphones have *tri-axis accelerometers*, which makes them able to measure the acceleration force over the device in three different axes.

Given this, accelerometers have become the preferred choice for continuous, unobtrusive and reliable method for human movement detection and monitoring, and with that, the use of accelerometry as a quantitative measure to completely define the movement of a body in space has many advantages over other existing techniques. [9]

When considering the use of accelerometers to measure the motion of a human body, certain factors need to be taken into account [10]: the **placement** of the device near the user (e.g. waist, chest, among others), how the user places the device **orientation** (e.g. face down, tilted, among others) and the **activity** being performed by the subject.

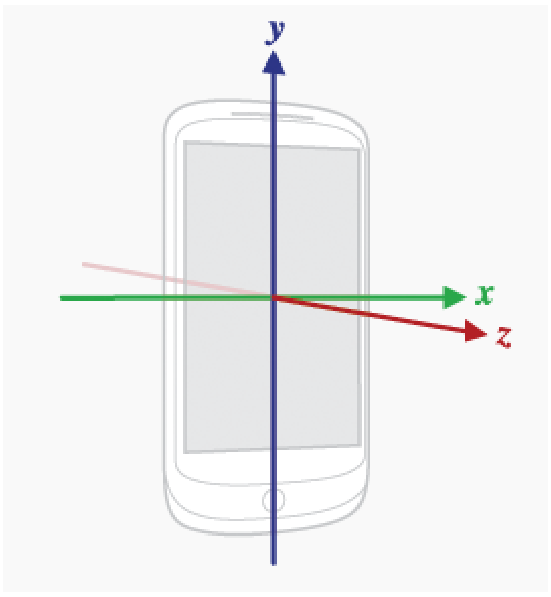


Figure 2.2: Accelerometer axis positioning in a smartphone [11]

In order to obtain the best results while measuring a person's attitude and gait, it is recommended that the accelerometer, and in the case of this work, the smartphone should be placed next or as closely as possible to the center of mass of the body, such as the sternum, hip or waist, although it is important to achieve such an algorithm that can determine a person's steps without considering a fixed position for the equipment. [10]

2.3.2 Gyroscope

The gyroscope is a sensor used for measuring or maintaining orientation, based on the principles of angular momentum. Gyroscopic sensors are used in navigation systems and for finding the position and orientation of devices. Combining a gyroscope with an accelerometer allows the device to sense motion on three axes - the roll (rotation around the YY axis), pitch (rotation around the XX axis) and yaw/azimuth (rotation around the ZZ axis) rotations allowing for more accurate motion sensing abilities.

MEMS gyroscopes are tiny masses on tiny springs, but instead of measuring acceleration, they are designed to measure a different force, the Coriolis force (the tendency for a free object to veer off course when viewed from a rotating reference frame) due to rotation. In the MEMS world, the gyroscope works by pushing a tiny mass back-and-forth along one axis. When the gyroscope is rotated, the Coriolis force makes the mass veer away from the direction it was vibrating, and it starts to move along a different axis. Movement along this new axis is sensed electrically, using capacitor plates: one capacitor plate is fixed to the frame and one is fixed to the moving mass. The Coriolis force acts only when the device is rotating, therefore gyroscopes measure only angular velocity, or, the speed at which the device is rotating. When the device is stationary, regardless of which direction the device is pointing, all three axes of the gyroscope will measure zero. [12]

The coordinate system is the same as the one used in the accelerometer. Rotation is positive in the counter-clockwise direction (right-hand rule). That is, an observer looking from some positive location on the x, y or z axis at a device positioned on the origin will report positive rotation if the device appeared to be rotating counter clockwise. The range will be at least 17.45 rad/s (i.e.: $\sim$ 1000 deg/s).

2.3.3 GPS

A Global Positioning System (GPS) is a space-based satellite navigation system that provides location and time information under any weather condition, anywhere on Earth as long as there is an unobstructed line of sight to at least four or more GPS satellites. The system provides critical capabilities to military, civil and commercial users around the world. It is maintained by the United States government and is freely accessible to anyone with a GPS receiver.

A GPS receiver calculates its position by precisely timing the signals sent by GPS satellites high above the Earth. Each satellite continually transmits messages that include the time the message was transmitted and the satellite position at time of message transmission. The receiver uses the messages received to determine the transit time of each message and computes the distance to each satellite using the speed of light. Each of these distances and satellites' locations defines a sphere. The receiver is on the surface of each of these spheres when the distances and the satellites' locations are correct. These distances and satellites' locations are used to compute the location of the receiver using the navigation equations. [13]

2.3.3.A Structure

The current GPS consists of three major segments: the space segment (SS), a control segment (CS), and a user segment (US). The U.S. Air Force develops, maintains, and operates the space and control segments. GPS satellites broadcast signals from space, and each GPS receiver uses these signals to calculate its three-dimensional location (latitude, longitude, and altitude) and the current time. The space segment is composed of 24 to 32 satellites in medium Earth orbit and also includes payload adapters to the boosters required to launch them into orbit. The control segment is composed of a master control station, an alternate master control station, and a host of dedicated and shared ground antennas and monitor stations. The user segment is composed of hundreds of thousands of U.S. and allied military users of the secure GPS Precise Positioning Service, and tens of millions of civil, commercial, and scientific users of the Standard Positioning Service. [13]

2.3.3.B GPS usage in Mobile and Smartphones

The majority of GPS receivers are built into mobile phones nowadays, with varying degrees of coverage and user accessibility. Some phones use an assisted GPS (A-GPS) function when they are out of range of their carrier's cell towers (although it gives a poor precision). Others can navigate worldwide with satellite GPS signals as well as a dedicated portable GPS receiver, upgrading their operation to A-GPS mode when in range.

2.4 Academic Development - State of the Art

Since 1980, several designs have been proposed to achieve a reliable navigation system for the visually impaired people in a way that it is possible to provide the best possible guidance while navigating through a certain region with the usage of various feedback signals (e.g. speech and/or vibration motors). To be able to provide a decent solution, it is necessary to better comprehend the aspects described below:

2.4.1 Positioning Techniques

In [14], it is suggested to first comprehend the ways of **human way-finding**: sensing the immediate environment for obstacles and hazards or navigating to remote destinations beyond the immediately perceptible environment and also to understand the concept of useful **navigation**: updating one's position and orientation while travelling to the desired destination, (with respect and accuracy), usually along some intended route.

As referred in section 2, navigation guidance systems based on simple GPS devices suffer from some positioning precision issues. To solve this, solutions have been proposed using Differential Global Positioning System (DGPS) and Real Time Kinematic (RTK) techniques [15, 16]. The use of Differential Global Positioning System (DGPS) implies the use of significantly more expensive equipment and more importantly the existence of a reference point, which is a huge obstacle to this work. Because of that, newer solutions were proposed where GPS was only used as part of the guidance system by exploring another positioning technique for the context of our target audience. Although the usage of the techniques stated above would lead to high development costs (mainly due to the needed frequency receivers), recent studies have proposed the use of such positioning techniques using low-cost single frequency receivers [17], for instance, in [18], a track estimation and simplification algorithm for visually impaired people was developed considering only the GPS available in a mobile device. Once a track (of a course) is loaded on the device, the visually impaired person was asked to try to walk with the device in order to help him move through that course without hesitation. If the person's position did not match one of the positions loaded on the dataset, it would alert and advise to correct its position. It was proven that the algorithm was able to detect the vast majority of steps taken by the person, keeping track of the course taken and also correcting possible deviations.

The algorithm and approach given by [18] encourages to keep considering the GPS as part of the positioning solution for the proposed system, since it was developed taking in consideration visually impaired people and also outdoor places - which are suited to part of our desired criteria. However, it demands a pre-loaded track into the system, something that is preferable to avoid while developing the system, in order to use less resources of the chosen device (mass storage usage in an application leads to a major battery consumption).

It is also important to, for the purpose of this work, establish the concept of velocity-based navigation that is usually referred to as **Dead Reckoning (DR)**. This technique relies on signals that indicate if a person is moving, through a velocity vector, in order to determine its new position based upon known or estimated speeds over the elapsed time and course. [19]. Also, and as an extension of the DR technique, it is important to establish the concept of **Pedestrian Dead Reckoning (PDR)** [20], which allows to collect the distance travelled since a person's starting point - explained in 2.4.2.A. That distance can be obtained with the usage of an accelerometer, but it is necessary to comprehend which region of the human body is best to place the device in order to obtain the best results as well as comprehend the various motion activities a person can achieve. All that must be considered with what was stated in 1.3, which is to develop such a system that is able to apply data fusion between GPS and inertial sensors (that apply the DR techniques), leading to a improvement of navigation and precise positioning using low-cost GPS receivers.

2.4.2 Activity Classification

In order to to reasonably apply the positioning techniques stated above, it is important to classify and understand the possible movements/motion activities a person is able to do (as illustrated in 2.3).

As described in [21], during walking and/or running stages, a person's body goes through several motion phases or states described as *gait* - the way locomotion is achieved using human limbs. With the usage of an accelerometer, it is possible to identify when a person takes a step, more precisely, the

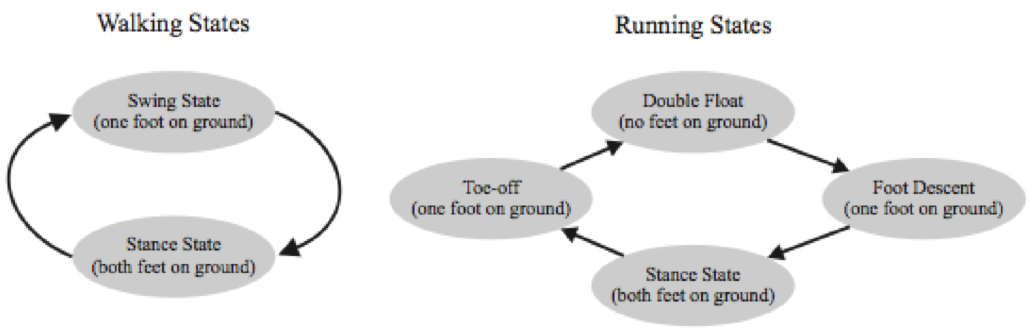

Walking States
Swing State (one foot on ground)
Stance State (both feet on ground)
Running States
Double Float (no feet on ground)
Toe-off (one foot on ground)
Foot Descent (one foot on ground)
Stance State (both feet on ground)

Figure 2.3: Human motion schematics [21]

simplest feature to identify is when a foot first hits the ground as this creates peaks in the acceleration record. However, the characteristics of a person's gait can change with the speed of motion, injury, age, disease or the surface where the motion takes place. These factors have also to be taken into account.

Regarding the speed of motion or *step frequency*, it is also important to be able to detect a person's running stage, not disregarding the fact that this stage can generate *false steps*, since the accelerometer can register peaks (equivalent as a step) during running motions (for instance, body swing). Hence, as suggested in [22] and as referenced in 2.3.1 , the smartphone will obtain the best results, i.e, the lowest erroneous results, if it is placed closely to the waist or hip.

2.4.2.A Step Amount and Length Estimation

In order to better understand the detection of a step given by a user (through the built-in sensors of the device) - the basics on a PDR algorithm, we must direct our focus to the device's accelerometer. As stated in 2.3.1, the user should place the device near the waist in order to obtain the best readings, so it can be further translated or interpreted into a sinusoidal graph. What we hope to collect is the amount of peaks detected in the obtained dataset (for offline analysis), but also in real-time, (such algorithm is explained in 2.4.2.B), to determine the distance travelled between two steps (step length). This obtained distance will help us if the GPS receiver suffers from a temporary loss of signal.

As stated in [20], after filtering the data (by removing the noise from the motion and removing the gravity component), we are able to determine properly the amount of steps and also the length between them, even regarding dynamic motion, such as running. Step length must have in consideration the height, weight and gender of the user.

2.4.2.B Activity Detection Algorithms

In [23], various smartphone oriented step and walk detection algorithms are evaluated. Due to the phone's specifications and its built-in sensors' limitations, it was determined that the ***Windowed Peak Detection - (WPD)*** is the optimal option for step counting regardless of the smartphone placement allied to a straightforward thresholding of the accelerometer standard deviation that will robustly and cheaply detect periods of walking.

The WPD algorithm establishes a threshold regarding the data collected from the calibration phase (due to the athlete's physical characteristics). Once that threshold is defined, a step will be considered and accounted if a peak that is detected surpasses that threshold. However, it needs to be further explored in order to develop an extension to this algorithm so it can adapt for any motion stage the person is in (whether it is a walking or a running stage).

2.4.3 Available Systems

There are also various systems developed (in an academic environment) with the goal of providing a reliable navigation solution suited for the visually impaired people, mostly focusing on the PDR techniques. In [24], a comparison of the available systems is made pointing out the strengths and weaknesses of each one, however, and for the purpose of this work, there are two systems that are relevant. Both of them are described as follows:

2.4.3.A Padati

In [25], the Padati solution is presented as an indoor navigation system relying on PDR techniques in order to detect the user's steps and respective length, not by relying on the device's location and orientation features but by using map matching and signal/particle filtering. Though the system has shown very promising results, the fact that it demands a manual and long calibration phase per each user and the usage of map matching features makes this system not suited to be considered for our solution.

2.4.3.B AutoGait

AutoGait, as presented in [26], is a mobile platform that determines a user's walking profile and estimates the distance walked using the GPS built-in receiver of the device. This platform was designed taking in account both indoor and outdoor scenarios and is able to calibrate to best suit the user's physical characteristics as a background process. Due to their GPS filtering and activity profiling algorithms, AutoGait achieved very good results, however, it lacks an orientation module (something crucial to our target audience) and it works on a separate device from the mobile phone, so it is not suited to be considered for our solution.

2.5 Commercial Applications & Devices

Herein the available mobile positioning and step counting applications are also considered, regarding the treatment of data obtained from the smartphone's sensors:

2.5.1 Pedometer & Navigation Applications

As suggested in [27], it is important to evaluate the accuracy of the available pedometer applications and appropriate ways to wear the cell phone when using pedometer applications. In this study, the tests were performed on a flat straight road and with the smartphones from various manufacturers and with different MOS placed at three different smartphone positions.

The tests suggest that most of the apps showed a low accuracy and high standard deviation in estimating the number of steps. Most apps estimated a higher number of steps when placed in the pocket of the pants compared to the other smartphone positions. This indicates that this position generates extra movement of the smartphone, which is interpreted as extra steps. Regarding devices with the Android, it was verified that the step estimation was influenced by the device itself and its respective sensors.

There are various applications that have positioning features like GPS monitoring by tracking the travel distance for each session or the usage of the accelerometer to get important vital statistics of your progress, but perhaps two of the most complete tracking applications available for Android is *Nike+ ™ Running*, *Strava ™ Run* and *PocketNavigator*:

2.5.1.A Nike+ ™ Running

Nike+ ™ Running [28] is an application developed for athletes or for running enthusiasts, it can track one's run (distance travelled, course taken, among some other features like distance, pace, time and calories burned) with a very good accuracy providing as well some audio feedback with informations stated. It provides a continuous and real time tracking feature, giving the possibility to trace the route on a map (by visual feedback). However, the main disadvantage of this application is the fact that it wasn't designed for the visually impaired, i.e., it is not prepared to guide the athlete (give directions) and it does not provide the needed feedback for the target audience of this work.

2.5.1.B Strava ™ Run

Strava ™ Run [29] is an application that is able to track the athlete's runs through the usage of the smartphone's GPS with a fair amount of accuracy, laying out the routes, times, and elevation of the user. Like Nike+ ™ Running, it provides somewhat useful information like elapsed time, distance and average pace along with the elevation change for each run. It is also able to connect with external devices (e.g. heart rate monitors) to provide a better informational feedback about the athlete's status and it can also full statistical readouts of your run. Once an athlete input its weight, age and height (factors that influence the obtained results), it will come up with calories burnt, average speeds and elements such as elevation, all mapped out in easy-to-read graphs.

As well as Nike+ ™ Running, the main disadvantage of this application is the fact that it wasn't designed for visually impaired, i.e.,it is not prepared to guide the athlete (give directions) and it does not provide the needed feedback for the target audience of this work.

2.5.1.C PocketNavigator

PocketNavigator adds tactile feedback to a simple but robust map-based navigation system that runs on any Android smartphone. The users can leave the device in their pockets, while being guided non-visually through vibration cues. [30] Although it provides the desired guidance and feedback, Pocket-Navigator is for navigation purposes only, it is not prepared for face pace activities, such as running sports.

2.5.2 Pedometer & Navigation Devices

As a fully developed service and system, **Drishti** [15] uses a precise position measurement system to guide blind users and help them travel in familiar and unfamiliar environments independently and safely. In outdoor locations it uses Differential Global Positioning System (DGPS), as stated in 2.4, as its location system to keep the user as close as possible to the central line of sidewalks. It provides the user with an optimal route by means of its dynamic routing and rerouting ability. The user can switch the system from an outdoor to an indoor environment with a simple vocal command and can get vocal prompts to avoid possible obstacles and step-by-step walking guidance to move about in an indoor environment. However, this system has several components that need to be attach to the user, making it uncomfortable to wear during long walks and unbearable or even impossible to use by an athlete. As future work, the developers of this system are considering the transition of the whole system to one single device, a smartphone.

On the particular use of guidance systems to sports, very few systems can be found. One of these systems is the TheSoundOfFootball, sponsored by Pepsi™[4], a project in which, using a stadium, they equipped with an array of cameras used to obtain a 3D scene of the football game. The processing of this scene allows sending sound commands to the user via an iPhone equipped with a digital compass. In the second system, proposed in [31], an indoor positioning system is developed using ultrasound satellites positioned in the four corners of the room. The athletes use an ultrasound emitter of 40KHz placed in a headband. Each emitter has an associated ID that is used by a centralised system to determine the position of each athlete. Each athlete is informed via a RF link of the location and proximity to other athletes by four vibrating actuators placed in a belt. No further details on the room size, precision, response time, or maximum number of athletes are given. Although interesting, this system requires a significant investment in the adaptation of the sporting room and in the equipment.

2.6 Summary

Throughout this chapter, various technologies are presented and revised for the purpose of this work, which is the guided navigation of visually impaired people. Alongside with those technologies are the multiple algorithmic approaches to best suit this scenario. The PDR algorithms are to ally with the device's components (mainly the positioning and the orientation sensors, as well as the GPS receiver) to better detect a person's movement and navigating it the best way possible.

However, the academic and commercial solutions tackle different aspects for the established scenario and there isn't available any (all-in-one and affordable) solution for the people that are affected with this disability. That doesn't mean that the work done until now isn't useful, on the contrary, those pioneer steps will help us lead to a more combined solution that will achieve the target audience's requirements.

[4]Source: `http://ourwork.se/thesoundoffootball/` - Last Visited: 10/2014

3

System Architecture

Contents

After analysing all of the state of the art technologies regarding our established requirements, herein is presented a solution that gathers the most suited positioning techniques out of the ones described in 2.4 into a device with built-in sensors that are able to track the position of an individual. Furthermore, guidance features are also added in order to make sure that the visually impaired athlete stays on course. In order to achieve such a system, in this chapter, the architecture of the presented solution is described, explaining each component's role to achieve the defined goals. Also in this chapter, some additional considerations are specified in order to explain certain approaches while developing the system, namely the athlete's biomechanics and the scenario where the system will be used, the running track.

3.1 System Design & Features

The proposed solution, named **TSAG - Tracking System for Athletic Guidance**, will be composed and developed as illustrated in figure 3.1:

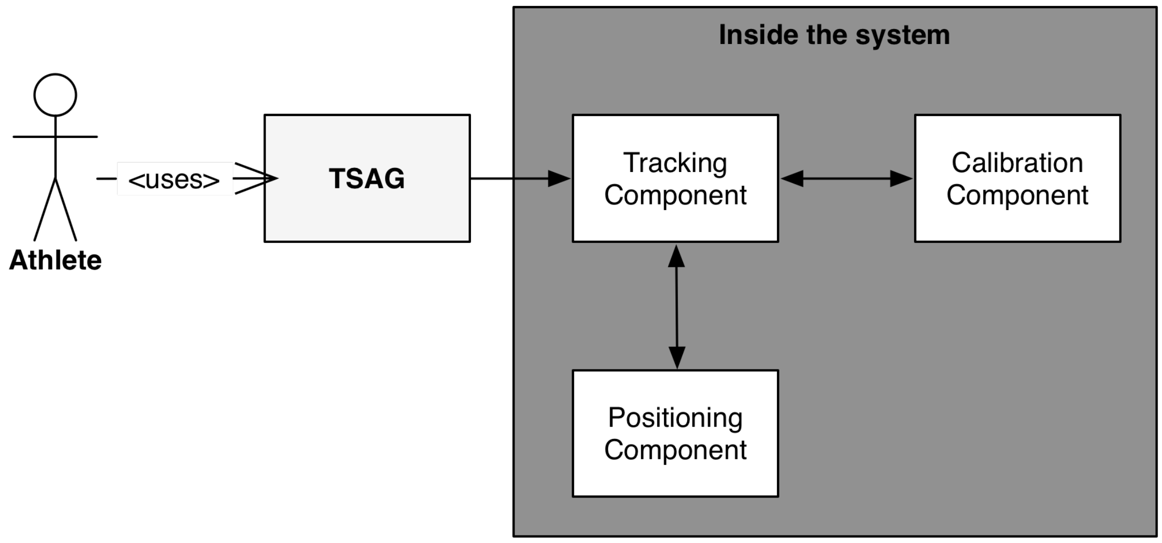


Figure 3.1: TSAG system architecture

The athlete will place the smartphone with the **TSAG** application running near its waist, since it provides the best motion readings/values [22]. The TSAG system will be composed by the **Tracking** component (the main one), which will allow the system to guide the athlete during its trial by evaluating its gait and its direction due to the usage of the gyroscope, which will evaluate the athlete's body rotations (through taking them into account) in order to detect possible deviations from the original course. Alongside, a **Calibration** component will measure the user's features (by the data obtained from the built-in sensors) and will create a user's profile or threshold in order to provide more accurate readings, especially when counting the athlete's steps and distance travelled. Meanwhile, a **Positioning** component, mainly featured by the device's GPS receiver, will provide the athlete's current position (in geodetic coordinates) as a reinforcement of the main component.

TSAG would also provide feedback to the athlete (vibratory or auditory aid) in several cases: **i)** the start and completion of a trial (considering the distances determined in the trials and by calculating the distance tracked), **ii)** mispositioning - the system will generate guidance information to allow the athlete to get back on the correct track. Additionally, the obtained information can be logged to monitor the athlete's performance in order to be able to improve it.

3.1.1 Tracking Component

The main component of the **TSAG** system, where the athlete will be able to navigate and be guided during the race. Once the athlete is ready to run along a track, the system will obtain the positioning values from the sensors mentioned in section 2.3 and log the athlete's course, as described in Figure 3.2.

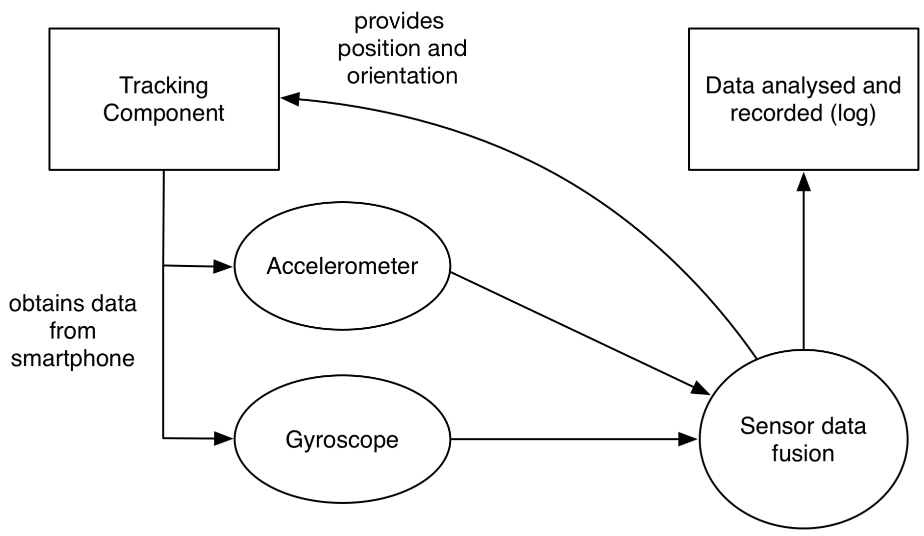

Tracking Component
provides position and orientation
Data analysed and recorded (log)
Accelerometer
obtains data from smartphone
Gyroscope
Sensor data fusion

Figure 3.2: Tracking Component

Since very precise feedback is needed, the system must detect if the athlete deviates from its course. If that happens, the system will inform of such event. In order to make this all possible, it is necessary to deal with a sensor data fusion algorithm that is able to detect the body's rotation during movement. A sensor data fusion algorithm can be developed in such a way that it is possible to obtain positioning and environment events by combining the data of two or more sensors. For the purpose of this work, we apply such algorithm to the device's accelerometer and gyroscope built-in sensors, where the data of the accelerometer can be used to calculate the athlete's travelled distance and the gyroscope to measure the rotations the athlete's body has made by matching the rotations that the device has suffered.

Algorithm 3.1: TSAG Sensor Data Fusion Algorithm

```
begin
    $accData \longleftarrow data\_provided\_by\_the\_accelerometer$
    $gyroData \longleftarrow data\_provided\_by\_the\_gyroscope$
    if $accData \neq null$ then
        $accelDetector()$
    else if $gyroData \neq null$ then
        $gyroFunction()$
```

The function/method (*gyroFunction()*) specified in the algorithm above is where the actual sensor fusion takes place. The data provided by the device's gyroscope is in the world's frame of reference (or the world's coordinate axis system), being necessary to adapt to the device's local frame of reference so that the obtained values match and express an actual rotation of the athlete's body. So the raw gyroscope values are fused with the ones obtained by an auxiliary sensor that measures the surrounding magnetic field, allowing to remap the coordinate system to the desired one.

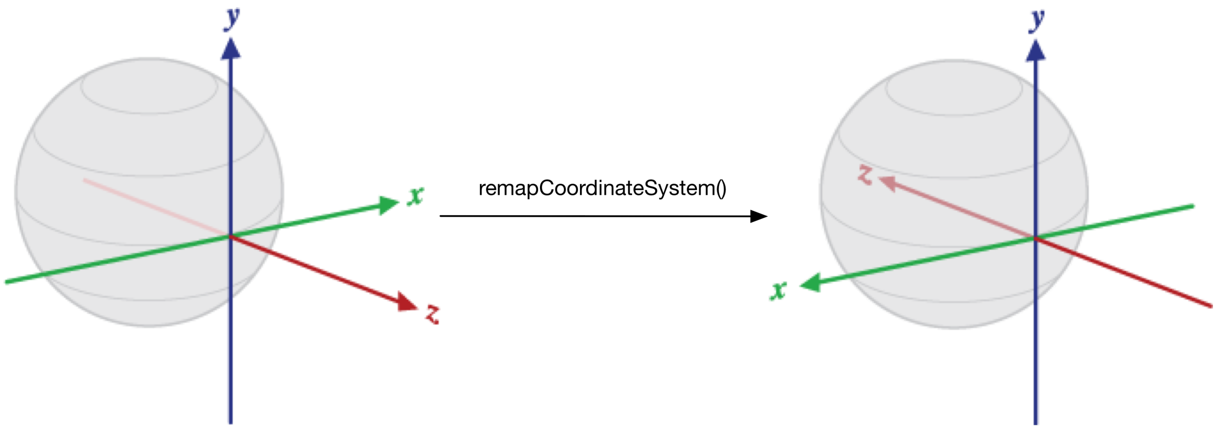


Figure 3.3: Remapping the coordinate system: for the world's frame of reference (left) to the desired frame of reference (right) [11]

3.1.2 Calibration Component

The system has to "learn about" its user, regarding the athlete's physiologic features such as weight and height, with the calibration system described in Figure 3.4. The system first collects the "raw" data (mainly regarding the accelerometer log) and establishes a threshold in order to determine when the athlete actually took a step. For specification and illustration purposes, this component is described as a sole component, but it actually acts as a background procedure of the Tracking component, mainly because both components are targeted for the same sensors (and want to obtain the same data). Nevertheless, it is necessary to always make a few test runs to adapt the system to the athlete, and, as suggested in [24] and other systems presented in 2, it is recommended that the athlete performs this calibration phase in a running track (with a flat surface) and the athlete's trainer or runner guide should accompany him during this phase.

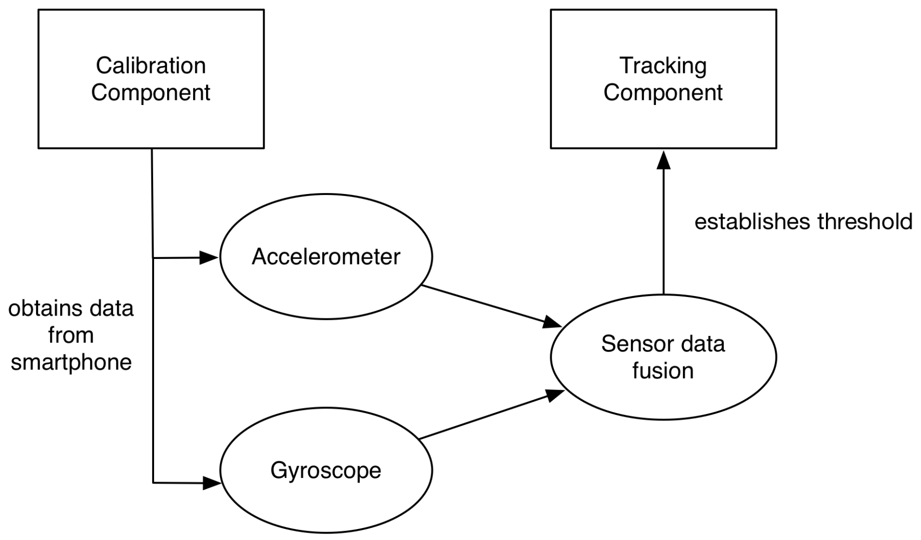


Figure 3.4: Calibration Component

3.1.3 Positioning Component

Since the considered scenario is in an outdoor location, the Positioning component can take full advantage of the smartphone's GPS receiver to obtain the athlete's current position, aiding the remaining components with the navigation and guidance features by providing a more accurate fused dataset. It is necessary to obtain at least a first positioning datum from the GPS receiver to establish as a reference but, after that datum is obtained, during the race and in case of loss of GPS signal, the PDR algorithm developed for the system can manage to keep track the athlete's motion and positioning (since the tracking component calculates the distance travelled) from the last obtained signal.

Both GPS and PDR can work separately or alongside one another, and the latter case is important to match positioning values when the GPS receiver recovers the signal so the system can check if the positioning values obtained from both parts have some type of gap between them. Despite that, the GPS will be the primary positioning source/provider to the system.

Algorithm 3.2: TSAG GPS and PDR algorithm

```
begin
    $accData \longleftarrow data\_provided\_by\_the\_accelerometer$
    $gpsData \longleftarrow data\_provided\_by\_the\_GPS\_receiver$
    if $gpsData \neq null$ then
        $obtainPosition()$
    if $accData \neq null$ then
        $distanceTravelled()$
```

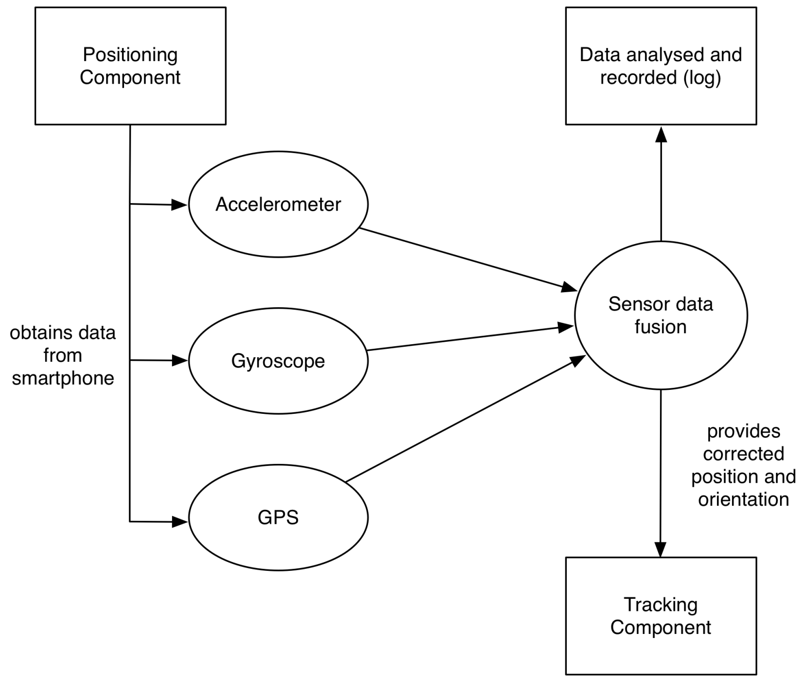

Positioning Component
Data analysed and recorded (log)
Accelerometer
Sensor data fusion
obtains data from smartphone
Gyroscope
provides corrected position and orientation
GPS
Tracking Component

Figure 3.5: Positioning Component

3.2 Other Considerations

The system presented is mainly focused on the three components described above, however, there are some additional considerations that are necessary to take into account so it can fully function for its purpose. These considerations are focused towards the user - the visually impaired athlete and also the scenario where the race takes place, the running track. Although it will not affect the general design of the system, these considerations will allow to better comprehend the necessities involved for this work, allowing it to best suit the user's needs and performance.

3.2.1 Biomechanics

A – Athlete's Running Speed It is important to characterise the subject of our work, the athlete. Even being visually impaired, the athlete has a high physical endurance and stamina, being able to perform harsh or demanding physical activities. In [32], a study is made to the biomechanics of a physical activity, such as running, and the impact it has on the athlete.

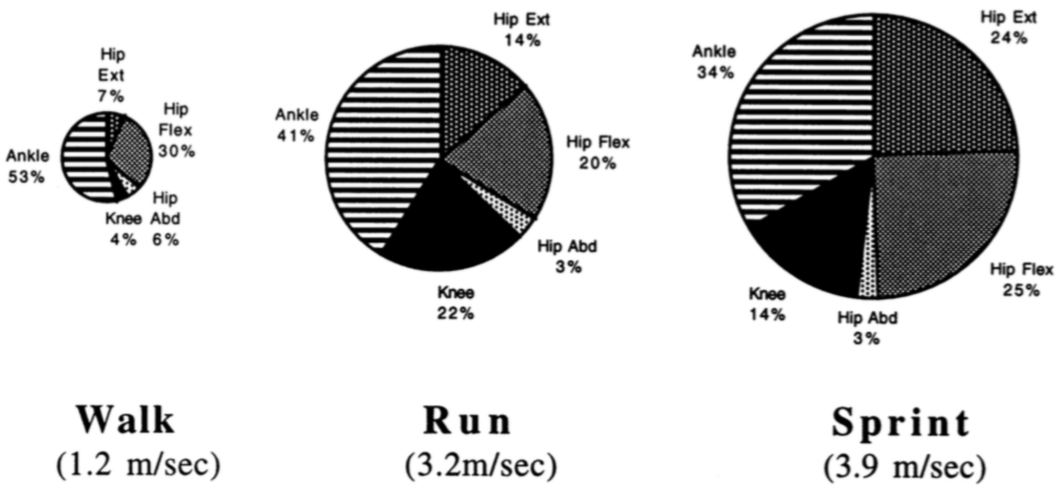


Figure 3.6: Body energy sources when performing an activity. [32]

It is established that, in average, an athlete will achieve running speeds between 3.2 m/s (11.52 km/h) and 3.9 m/s (14.04 km/h).

B – Step Length As stated in 2.4.2.A, in order to develop a proper PDR algorithm, it is necessary to know the height, weight and gender of the user, since they are factors that can alter the output results. In [20], it is proposed a kinetic formula (static approach) to best obtain the stride (or step length):

$$step\_size = height.k \text{ [20]}$$

where k is the kinetic factor and the height is measured in meters. The static approach can be considered for this system because although it is an intense physical activity, when the athlete is running, its stride does not vary during the race. In fact, is important to maintain the same pace during a race. **On average, an athlete's step length is between 1.00m to 1.20m.**

C – Step Period If we consider the results obtained above, it is possible to obtain the athlete's average step period:

Table 3.1: Step Period Estimation (Minimum and Maximum)

Minimum Step Period (s):	$\frac{1m}{3.9m/s} = 0.2564$
Maximum Step Period (s):	$\frac{1.2m}{3.2m/s} = 0.3750$

This data allows to obtain the rate of each step occurrence, since $F(Hz) = \frac{1}{T(s)}$:

Table 3.2: Step Occurrence Rate (Minimum and Maximum)

Minimum Rate (Hz):	$\frac{1}{0.3750s} = 2,666$
Maximum Rate (Hz):	$\frac{1}{0.2564s} = 3,900$

So, and by Nyquist–Shannon's sampling theorem[1], which tells that in order to not lose any important data when sampling a signal (in this case the data provided by the sensor), it is necessary to sample it at least two times the value of its bandwidth: $F \geqq 8Hz$.

3.2.2 Scenario Specifications

Another important consideration to take into account is the scenario in which the system will be used - the running track. As described in figure 3.7 and in more detail in B, the paralympic athletes with visual impairments use the standard running tracks regulated by the **IAAF** - International Association of Athletics Federations.

A key aspect to focus on is the semi-circle (right after the 84.39m of a straight section) due to the fact that it is necessary to know the angle/deviation that an athlete needs to make to stay in course. It is known that:

Inner_Circle Radius $= 36.5m$ and **Outer_Circle Radius** $= 45,04m$

Table 3.3: Course Deviation on the Semi-Circle (Minimum and Maximum) - in degrees

Minimum Deviation:	$\sin(\theta) = \frac{1.10}{45,04} \approx 0.0244253 \| \theta \approx 1.4$
Maximum Deviation:	$\sin(\theta) = \frac{1.10}{36.5} \approx 0.0301402 \| \theta \approx 1.7$

meaning that it is necessary to consider the **minimum deviation value of 1.4 degrees** so the athlete stays on course. If the athlete starts to form a wider deviation, the system has to alert of such event and has to help the athlete to get back on track. However, it is necessary to take into account the hip

[1]Source: Princeton University - `http://goo.gl/CYdmJW` - Last Visited: 10/2014

rotation that the body makes while in motion and when it's about to perform a turn. In [33], several hip rotations were measured while the subjects were performing physical activities such as walking and running. As an average result, it was established that the hip rotation angle is ≈ 15 to 20 degrees. So the minimum deviation that the **TSAG** system must perceive is the combination of the hip rotation angle with the minimum track deviation.

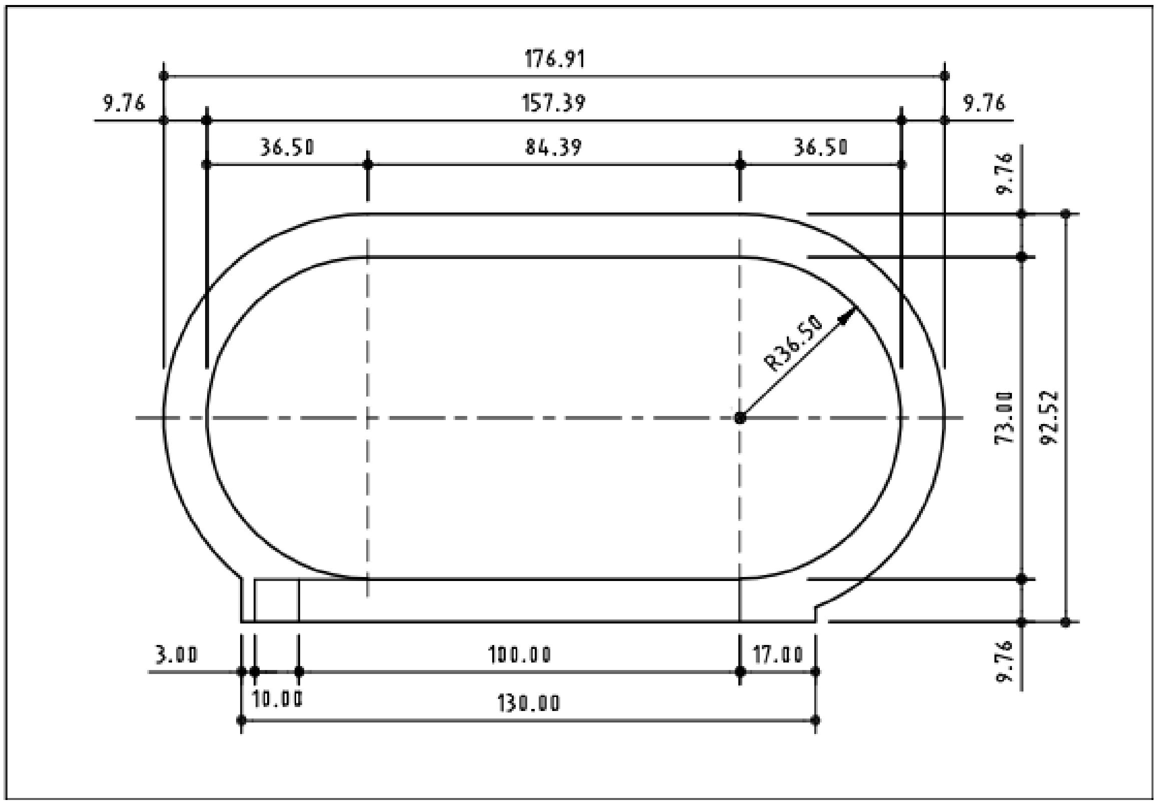


Figure 3.7: Shape and dimensions of the 400m Standard Track (Radius 36.50m) - all the presented dimensions in m [34]

3.2.3 Chosen Device

Due to the requirements established in 1.2, namely the choice of a device that is low-cost, easy to wear and highly efficient, with the integration of various sensors needed and described in 2.3, it is considered the development of the proposed solution on a smartphone.

For the considered device and due to its specifications, it is important to consider which are the most versatile and commonly available mobile operation systems. Recent studies have shown[2] that the **Android OS** is the most used MOS with 91% of market share, mostly due to the *open-source* factor, giving the ability to customise many aspects of the operating system. With such results, the solution proposed

[2]Source: StatCounter (Global Stats) - `http://gs.statcounter.com/` - Last Visited: 10/2014

will be firstly developed for Android. Once this version of the system is fully prepared and tested, it will be considered as future work the development for other MOS such as *iOS* and *WindowsPhone*, since they have the biggest market share next to Android.

The device must also have built-in accelerometer and gyroscope sensors in order to work with the orientation aspect of the system as well as a GPS receiver so it can provide positioning features to the positioning and navigation aspect of the system. The system could also work solely with an accelerometer and a GPS receiver but it would not provide the best results comparing with the combination former mentioned.

3.3 Summary

Throughout this chapter, the considerations needed for the development of the system were described and further explained. These considerations, although not entirely technical, are major factors to the conception of this system.

The system architecture, specified with the design and its features was presented: the **TSAG** system, as a **Tracking** component, which allows the system to guide and navigate the athlete, a **Calibration** component, which allows to better understand the system's user by analysing the sensor's data, and finally, a **Positioning** component, which takes advantage of the device's built-in GPS receiver to obtain fairly accurate values of the athlete's current position.

4 Implementation

Contents

With the system presented in the earlier chapter, mainly the components that are part of, it is necessary to understand how the techniques described in 2 get together and work along with the smartphone's built-in sensors and the GPS receiver. So, in this chapter, the implementation of the **TSAG** system will be described in a away that it makes it able to comprehend how everything gets together by making it suitable for visually impaired athletes. The description of the system (and its implementation) will be usage/user oriented, which means that each component will be described on how it functions for a certain feature.

4.1 Development Process

As stated above, the developed work is composed by a combination of available work adapted for this solution but it also has new features built specifically for it due to the targeted device and its specifications. All the algorithms, each with its own functions/methods were developed in *Java* - Android oriented.

4.1.1 Tracking & Calibration Components

In order to either detect or track the athlete's movement, the use of the built-in sensors such as the accelerometer and the gyroscope is needed. In order to access them, both the ***Sensor*** and ***SensorManager*** Android libraries or Application Programming Interface (API) are necessary.

The *Sensor* API consists of classes that are able to request and process information from the sensors built-in the device's hardware while the *SensorManager* is an Android system service that enables access to the hardware's sensors, making it possible for the application to register (and track) sensor-related events. [12]

In order to obtain data from the sensors, they have to be triggered at a specific frame rate which can be selected by a trigger mode:

Fastest: The sensors are triggered at a frame rate of $100Hz$ with a delay of $20000\mu s$

Game: The sensors are triggered at a frame rate of $50Hz$ with a delay of $20000\mu s$

Normal: The sensors are triggered at a frame rate of $\approx 15Hz$ with a delay of $60000\mu s$

UI: The sensors are triggered at a frame rate of $\approx 10Hz$ with no delay

Considering the biomechanics factors established in 3.2.1 and considering that the output of the sensors does not need to be displayed on the screen (hence the UI frame rate), the trigger modes chosen for this work will be the **Normal** mode to trigger the gyroscope and the **Fastest** for the accelerometer. The latter choice is due to the fact that, when using the accelerometer data to develop a proper step detector and counter algorithm (which is able to distinguish a shake - example of false step - from an actual step taken by the athlete), the chosen trigger mode is the one that provides best results.

4.1.1.A Peak Detection

Focusing on the ***Windowed Peak Detection - (WPD)*** algorithm mentioned in 2.4.2.B, which checks the peaks generated from the accelerometer output data, it has to be adapted to the running activity, which means the ”window” used to detect those peaks has to be flexible, or in some cases, non-existent. So, the approach presented in [35] is used but adapted to the system’s needs - algorithm in Listing 4.1.

A – Calibration : This is where the calibration component will take place. After collecting a reasonable amount of data a threshold will be found, meaning that the peaks detected in the algorithm are more reliable and feasible to be considered as a step.

```java
private void checkData(ArrayList<Pair> accelFireData, long timeStamp)
    {
        boolean stepAlreadyDetected = false;
        Iterator<Pair> iterator = accelFireData.iterator();
        while (iterator.hasNext() && !stepAlreadyDetected) {
            stepAlreadyDetected = iterator.next().first.equals(mLastStepTime);
        }
        if (!stepAlreadyDetected) {
            int firstPosition = Collections.binarySearch(mMagneticFireData, accelFireData.
                get(0).first);
            int secondPosition = Collections
                .binarySearch(mMagneticFireData, accelFireData.get(accelFireData.size() -
                    1).first - 1);
            if (firstPosition > 0 || secondPosition > 0 || firstPosition != secondPosition)
                {
                if (firstPosition < 0){
                    firstPosition = -firstPosition - 1;
                }
                if (firstPosition < mMagneticFireData.size() && firstPosition > 0){
                    mMagneticFireData = new ArrayList<Long>(
                           mMagneticFireData.subList(firstPosition - 1, mMagneticFireData.
                               size()));
                }
                iterator = accelFireData.iterator();
                while (iterator.hasNext()) {
                    if (iterator.next().second) {
                        mLastStepTime = timeStamp;
                        accelFireData.remove(accelFireData.size() - 1);
                        accelFireData.add(new Pair(timeStamp, false));
                        onStep();
                        break;
                    }
                }
            }
        }
    }
```

Listing 4.1: Peak Detection algorithm

4.1.1.B Step Amount and Distance Travelled

With the threshold established, all the peaks that the algorithm detects will be considered as steps. This means that we can collect and consider as valid all the steps that were detected by this algorithm during the race. With this amount, we can obtain the distance the athlete has travelled since the starting point. In order to do so, we use the following formula:

$$\textbf{num\_stride} = numSteps$$
$$\textbf{distTravelled(m)} = \frac{num\_stride * stride\_length}{100}$$

Filtering: Also, as stated in [20], the output acceleration data must be filtered in order to be gravity-free and noise-free (to measure the real acceleration of the device) so it can be less prone to detect false steps. To achieve this, the following filter was used:

```
private float HIGHALPHA = 0.8f;
public float[] accelFilter(float[] input, float[] output){
        gravity[0] = HIGHALPHA * gravity[0] + (1 - HIGHALPHA) * input[0];
        gravity[1] = HIGHALPHA * gravity[1] + (1 - HIGHALPHA) * input[1];
        gravity[2] = HIGHALPHA * gravity[2] + (1 - HIGHALPHA) * input[2];

        output[0] = input[0] - gravity[0];
        output[1] = input[1] - gravity[1];
        output[2] = input[2] - gravity[2];
        return output;
}
```

Listing 4.2: Filter

A – PDR factor : Between known GPS signals, the algorithm will also accumulate the distance. If by any chance the GPS receiver does not recover the signal, it will calculate an estimation of the GPS coordinate at the end position (this feature will be further explained in 4.1.2).

4.1.1.C Body Rotation & Orientation (Data Fusion Procedure)

The **TSAG** system must be able to detect the athlete's body rotation so it can predict possible track deviations, i.e., prevent the possibility of the athlete to get out of the track. This will be accomplished by using a data fusion procedure/algorithm that gathers both the accelerometer and gyroscope sensors and a software-based sensor that measures the magnetic field. The data fusion algorithm developed [36] is the "foundation" that makes the system able to tell the athlete's orientation.

In order for this data fusion algorithm to work, the sensors mentioned above have to be ready to provide data at the same time. To ensure that, the *synchronized* function is used, however, each sensor has its own handler. This handler (for the gyroscope) will need some accelerometer and magnetic field values prior to function. As stated in 4.1.1, the frame of reference has to be adapted in order to accurately read the rotations the body makes. This is achieved by applying rotation matrices to the values obtained from the gyroscope and the combined fused values of the accelerometer and the magnetic field sensors.

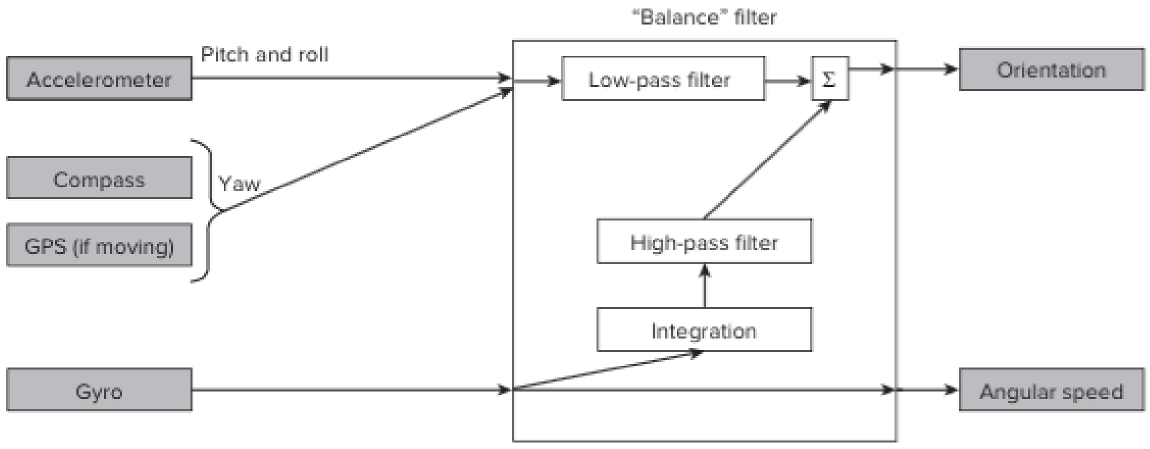


Figure 4.1: Example of SensorFusion - Balance Filter [12]

Although this is the best way to determine the rotations made by the athlete, when the gyroscope is used continuously (during a large period), it tends to provide faulty values due to integration issues. To correct that, a timer task was developed to be triggered with a fixed period to apply correction filters to the values obtained. The values that are the outcome of this task are the ones that matter for our work (example in Listing 4.3).

```
class calculateFusedOrientationTask extends TimerTask {
        public void run() {
            float oneMinusCoeff = 1.0f - FILTER_COEFFICIENT;

            // azimuth
            if (valuesOrientation[0] < -0.5 * Math.PI && accMagOrientation[0] > 0.0) {
                fusedOrientation[0] = (float) (FILTER_COEFFICIENT * (valuesOrientation[0] +
                     2.0 * Math.PI) + oneMinusCoeff * accMagOrientation[0]);
                fusedOrientation[0] -= (fusedOrientation[0] > Math.PI) ? 2.0 * Math.PI : 0;
            }
            else if (accMagOrientation[0] < -0.5 * Math.PI && valuesOrientation[0] > 0.0) {
                fusedOrientation[0] = (float) (FILTER_COEFFICIENT * valuesOrientation[0] +
                    oneMinusCoeff * (accMagOrientation[0] + 2.0 * Math.PI));
                fusedOrientation[0] -= (fusedOrientation[0] > Math.PI)? 2.0 * Math.PI : 0;
            }
            else {
                fusedOrientation[0] = FILTER_COEFFICIENT * valuesOrientation[0] +
                    oneMinusCoeff * accMagOrientation[0];
            }
        (...)
}
```

Listing 4.3: Fused Orientation algorithm - azimuth

With the proper values obtained, we need to consider which values can tell us if the athlete is deviating from its original track. If we consider the frame of reference in 4.1.1, mainly the ZZ and XX

axis, we can determine if a deviation occurs by normalising the current pair of values (azimuth, pitch) and creating a new pair with the current pitch only, as illustrated in Figure 4.2. From here, we use the Pythagorean theorem by forming a right triangle, and with that, it is possible to find the deviation angle by using the following formula: $\sin(\theta) = \frac{module\_current\_pitch}{module\_current\_pair}$

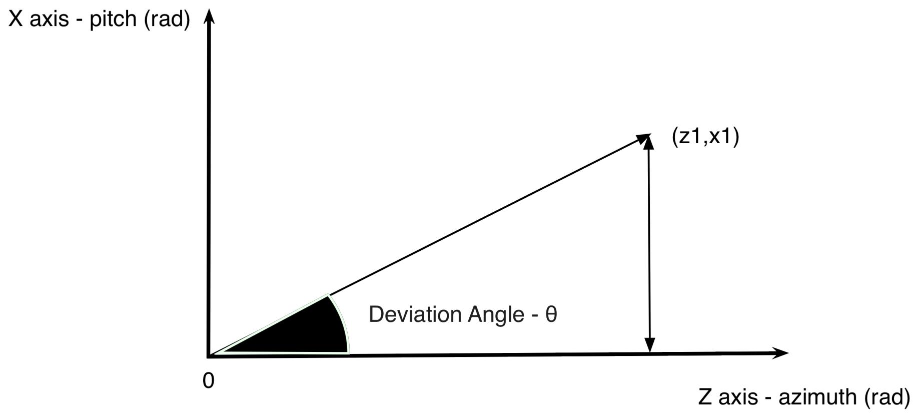


Figure 4.2: Example of a deviation

The final factor that is needed to take into account is that a sensor event is registered at $0.030s$ with an average step period, as established in 3.2, is about $0.300s$. So, we can only consider that a (possible) actual deviation is occurring after $min\_num\_sensor\_events \geqq 10$. Thus, in order to perceive if an actual deviation is occurring, the athlete should be alerted only after that number of events is registered (for each deviation) - algorithm in Appendix A.1.

4.1.2 Positioning Component

The **TSAG** system also has a positioning component, which allows us to obtain the coordinates of the athlete on the racing track. In order for this to be possible it is necessary to create an Android Service, which is able to work alongside the previous mentioned components (in the background). Every time the athlete achieves a new position on the racing track, the GPS receiver will capture a new coordinate set (more focused on the latitude and longitude), and each set is stored in a coordinate log. The implemented service was based on Rahul Wingnity's approach [37].

A – Usage Restriction: In order to take full advantage of the **TSAG** system, it is important to obtain a first valid position from the GPS signal (through the receiver). It will take a few minutes for the GPS receiver to obtain a constellation set and be able to provide new coordinate sets. When the receiver has finally obtained a new position, it will alert the system so it can provide a TTS aid informing the athlete of such event. Only after that first position, both the GPS and PDR components of **TSAG** can obtain or track the following athlete's positions.

4.1.2.A PDR algorithm:

As stated in 4.1.1, with the tracking component (that uses the accelerometer and gyroscope), we are able to collect the distance the athlete has travelled considering a first valid coordinate set, but in case of loss of GPS signal and the device's receiver cannot provide a position, the PDR algorithm is used to provide an estimate of the athlete's current position. This is possible to achieve by using Thaddeus Vincenty's [38] direct geodetic solution algorithm. The algorithm makes it possible to obtain a new coordinate set if a starting point coordinate set is provided and the distance that (in this case, the athlete has travelled) was made. In order to implement this algorithm in the Android environment, the solution provided by Mike Gavaghan [39] was used.

Combining the data that is obtained from the GPS receiver with the usage of Android's ***LocationManager*** library, when a coordinate set is obtained, the system will utilise the latitude, longitude and bearing (in case of loss of signal from the GPS) in the PDR algorithm. The *LocationManager* allows an app that needs to receive up-to-date location information, allowing to choose a *LocationProvider* - a source of location information, such as the GPS receiver, which will provide those requested location updates. A *Location* or a coordinate set in the Android context is a class that contains a certain location datum such as the latitude, longitude and altitude, among other parameters, but, in order to detect changes from the provider, or in order to detect a new location, it is necessary to have a *LocationListener*, which is able to detect any change from the *LocationProvider* [12].

Since the coordinate set was obtained through the **WGS84** reference ellipsoid, it is necessary to pass this information as an argument so that the new coordinate set will also be in that reference. The last argument, the distance, is provided by the tracking component. The combination of these five arguments used in the formula is the key of **TSAG**'s PDR approach and algorithm.

It is important to recall that the PDR only needs to work in case of loss of GPS signal, so it is only activated when that happens. The switching process between the two mentioned modes is automatic and transparent to the user. Although the PDR algorithm can replace the GPS receiver in the track field, it is only possible to do it during a few meters due to the track's characteristics. If, for instance, the GPS receiver was only able to obtain the starting position coordinate set, it will not be able to perceive that the athlete is running through the curve; in fact, the algorithm will consider that the athlete has been on a straight path, so it will not be able to provide the actual course the athlete has taken. However, the races will take place at an outdoor track field where the GPS signal will be available during the majority of the time and the PDR algorithm will work properly between short gaps of GPS signal loss.

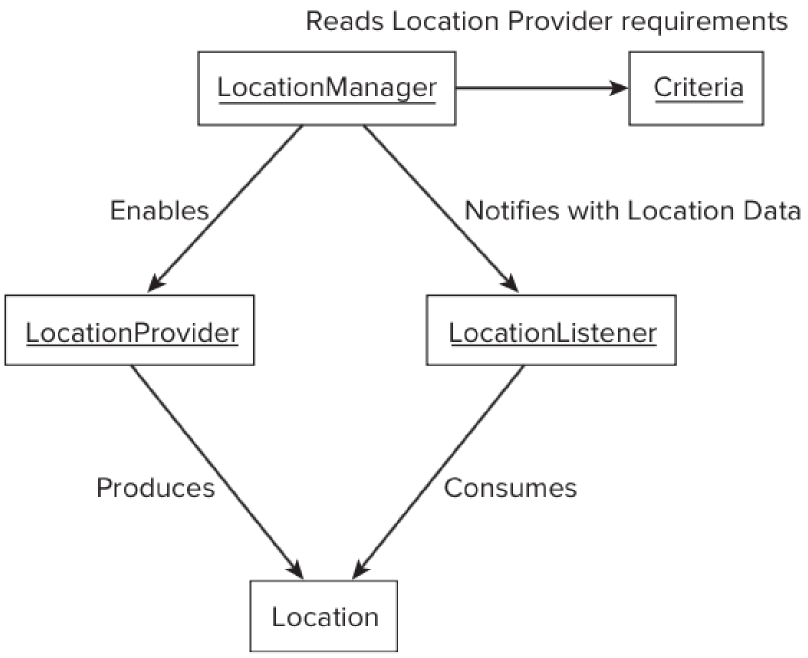


Figure 4.3: Location Components in Android

4.1.3 Feedback Provider

Due to the athlete's visual impairment, it is necessary to implement a resourceful and efficient feedback provider component that will be able to guide the athlete in a way that any auxiliary aid will not be needed during the race. This component will have a vibratory service and a TTS module.

The vibratory service (provided as an Android library) allows us to make the device vibrate accordingly with certain events that are considered important and relevant for the whole user experience. For this work, two vibratory patterns will be used, each one to be triggered when a deviation occurs (either to the left or to the right). Due to the high consumption of CPU resources (that have to be allocated in order for this service to work properly without disrupting **TSAG**'s main component), an auxiliary thread will be created for each vibratory pattern and can only be triggered when a deviation occurs. A vibratory pattern is simply a time sequence chosen to make the device vibrate when a certain event is detected.

```java
Thread tsag_vibRight = new Thread(new Runnable() {
        public void run() {
            try {
                if(v.hasVibrator()){
                    v.vibrate(50);
                }
            }
            catch (Throwable t) {
                Log.i("Vibration", "Thread  exception "+t);
            }
        }
    });
```

Listing 4.4: Example of a vibratory pattern

The TTS module will provide speech aids to the athlete whether they have an informative character (e.g distance travelled) or an imperative character (e.g. deviation detected - must correct position). This module is also provided as an Android library. The configuration is based on four factors: pitch, rate, language and sentence to be spoken. These factors, especially the first two, have to be manipulated in such a way that the final output can alert the user but not overwhelm it.

```java
@Override
    public void onInit(int status) {

        if (status == TextToSpeech.SUCCESS) {

            int result = tts.setLanguage(tts.getLanguage());
            if (result == TextToSpeech.LANG_MISSING_DATA
                    || result == TextToSpeech.LANG_NOT_SUPPORTED) {
                Log.e("TTS", "This Language is not supported");
            }
        } else {
            Log.e("TTS", "Initilization Failed!");
        }
    }
    (...)
    @Override
    public void onActivityResult(int requestCode, int resultCode, Intent data) {

        if (requestCode == TTS_CHECK_CODE) {
            if (resultCode == TextToSpeech.Engine.CHECK_VOICE_DATA_PASS) {
                // the user has the necessary data - create the TTS
                tts = new TextToSpeech(this, this);
            } else {
                // no data - install it now
                Intent installTTSIntent = new Intent();
                installTTSIntent.setAction(TextToSpeech.Engine.ACTION_INSTALL_TTS_DATA);
                startActivity(installTTSIntent);
            }
        }
        super.onActivityResult(requestCode, resultCode, data);
    }
    (...)
    tts.speak((String) "Example...", TextToSpeech.QUEUE_FLUSH, null);
    (...)
```

Listing 4.5: TTS Example

4.1.4 Continuous Tracking

Although some of the components that belong to **TSAG** are services that are able to run in the background (i.e. phone with screen off or locked), the main component that gathers all the data is developed as an activity. An Android activity has a lifecycle, which can be disrupted if it receives a termination/-

pause signal, meaning that some components can be put on hold, causing the **TSAG** system to not work properly. To prevent that from happening, a **Wakelock** is implemented (part of the *PowerManager* Android library) to lock **CPU** resources at all times if the activity (or in this case, the tracking component) is running.

```
(...)
PowerManager powerManager = (PowerManager) getSystemService(POWER_SERVICE);
          wakeLock = powerManager.newWakeLock(PowerManager.PARTIAL_WAKE_LOCK,
                    "MyWakelockTag");
(...)
```

Listing 4.6: Wakelock Example

4.2 Application

The application available for the user has to be installed in a device with the Android MOS and it needs to have an accelerometer, gyroscope and GPS receiver. Although the user won't be able to perceive this, the application will only display one screen with three buttons, each with its own specific function.

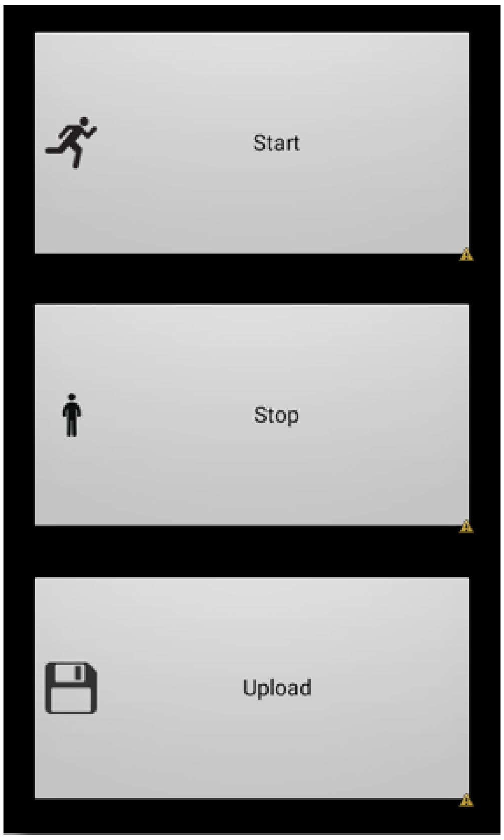


Figure 4.4: TSAG's main screen

The functions that will be more relevant for the user are the **Start** and **Stop** functions, where the Start function will initialise all of **TSAG**'s components in order to guide the athlete during the race, and the Stop function, which will put the components on hold until the next race.

There is also a third function available, but it is more useful for the athlete's trainer, which is the **Upload** function. When pressed, all the information collected from the races that the athlete has made will be stored in *.csv* values so they can be used for data analysis. All the functions with their respective buttons are placed in a way that the athlete is able to use without any aid.

The Start button is placed at the top of the screen, the Stop button in the middle and the Upload button at the bottom. Each button's dimensions are large enough to be perceived and pressed by people with visual impairments, but, in the particular case of the athlete, to be able to use this system, it only needs to perceive the three basic positions mentioned earlier.

If the easiness of use is not yet guaranteed, the athlete may also press each position of the screen and a TTS information command will tell which function the athlete is about to select. After the application has been prepared to track on the athlete's race, the device needs to be placed near the hip or waist to obtain the best values, as established in [22]. It is important to place it in a way such that the lock button won't be pressed to continuously track the trial.

4.3 Summary

In this chapter, all of **TSAG**'s components described in 3 were described regarding its implementation. Since it was developed for the Android environment, for the **Tracking** component, it was necessary to use the *Sensor* and *SensorManager* libraries or API in order to access the device's built-in accelerometer and gyroscope and retrieve each sensor's output data.

With the obtained data, it was possible to obtain certain characteristics such as the amount of steps the athlete takes as well as the distance travelled during the course. When the data of both sensors is combined (Sensor Fusion), the **TSAG** system is able to detect the orientation the athlete is taking at a certain moment during the race, also being able to detect possible deviations.

In order to provide location coordinates to the **TSAG** system, the **Positioning** component uses the library or API **LocationManager**, by which, with the GPS as the selected location provider, it is able to obtain a *Location* object with a coordinate set that is useful for the system. However, in case of loss of GPS signal, a PDR algorithm was also implemented in the system that is able to estimate the destination coordinate set if it is given a start position (coordinate set) and the distance obtained from the Tracking component.

The athlete will obtain all this information due to **TSAG**'s feedback provider, which is composed by a TTS module and vibratory patterns that will be triggered when an event of enough relevance to the athlete and its performance on the race occurs.

5 Evaluation

Contents

After the description of the **TSAG** system presented in 4, in this chapter, it is presented the evaluation to the system regarding some key aspects and it is presented a analysis of the obtained results. In order to properly test the system, a methodology must be followed and explained to infer proper credibility to the obtained results.

5.1 Methodology and Scenarios

The evaluation (or better yet, the methodology) prepared to test the **TSAG** system will be focused on each system's components in separate, but also the system will be evaluated as a whole, which means, it will be tested as the visual impaired athletes should experience the system. The tests will also focus on the activity that is being performed by the user who's testing the application.

A – Separate Test 1: The first separate component test will verify the **accuracy** of the tracking component, mainly the step counting (and amount) and also the distance estimation. To compare the amount of steps, the test user will be asked to control as best as possible the amount of steps that it has done during the chosen test course (while walking or running). The result obtained by the system will be compared to the amount the user has counted by itself. The distance calculated by the system will be compared to the distance provided between two known points (using *Google Maps™*).

B – Separate Test 2: The second separate component test will verify the **precision** of the **TSAG**'s positioning component. All the GPS data (obtained from the receiver) will be evaluated by tracing a route (with the coordinates log) on an online tool named *GPS Visualizer*[1] to verify the proximity to the real target and chosen course. The same will be done with the coordinate log obtained from the PDR algorithm. The logs will be obtained while walking and also running, as the first test mentioned above.

C – System Test: This test will evaluate the **TSAG** as a whole, but it will have special focus on the orientation and feedback components. Since that, in this test, the objective is to replicate the conditions that the athlete has, it is important to verify one of the main rules that the athletes have to follow while competing:

> ”Athletes in Sport Classes T/F11 - competitions with athletes that possess some type of visual impairment - must wear approved opaque glasses or an appropriate substitute during all track and/or field events. The opaque glasses or their substitute must be approved by the responsible technical official and must in their opinion be effective in blocking out sufficient light to compete fairly.” [40]

[1]Source: `http://www.gpsvisualizer.com/`: Last Visited: 10/2014

This means that the test user, if not visually impaired, will be asked to be blindfolded, or at the very least, to close its eyes while walking or running through the test course. Since it will be tested in an outside area, the user will also have to use a headphone set in order to listen to the TTS commands of the system. With this, it will be possible to evaluate the adaptation that the user has with the system and also the accuracy and precision (of the system and user response to it) by the orientation aids it will receive (by the system) in order not to deviate from the original test course.

5.1.1 Chosen Test Courses

One of the test courses chosen to evaluate the **TSAG** system is located in Fornebu, Akershus, Norway, between the coordinates $(59.8950084N, 10.6296923E)$ and $(59.8957185N, 10.6285701E)$, which means a 100 meters test (with a straight track) course in a flat surface. In the surroundings of this chosen course there are a few buildings in order to provide some difficulty to the GPS system to obtain signal and subsequent positions, forcing the PDR algorithm to work as much as possible.

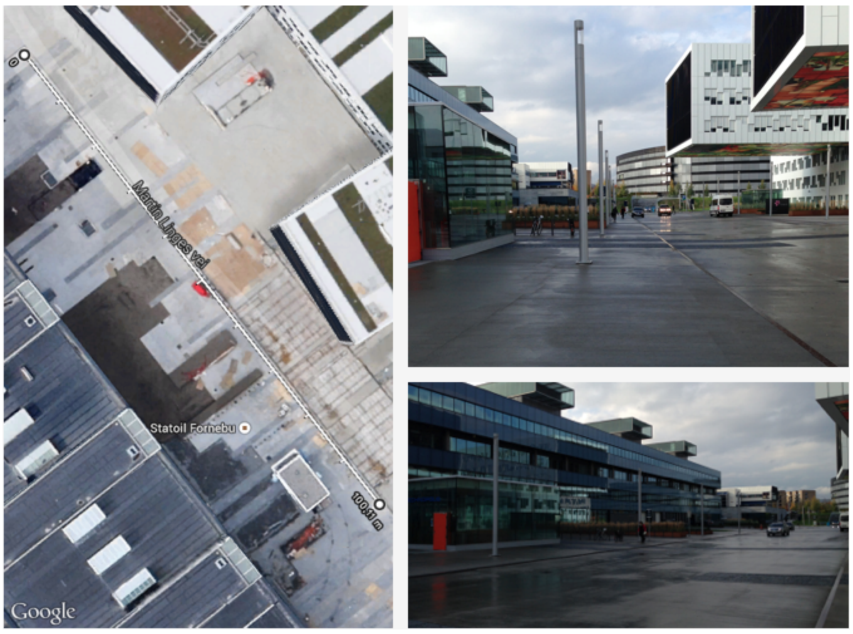


Figure 5.1: Scenario1: **left** - overview, **top right** - front view, **bottom right** - side view

The second test course to evaluate the **TSAG** system is also in Fornebu, Akershus, Norway, between coordinates $(59.8962722N, 10.6274282E)$ and $(59.8958057N, 10.6262629E)$ which means also a 100 meters test course. The purpose in this test and chosen course is to evaluate the adaptation of the system regarding the ability to perceive curves and deviations that the athlete may make while contouring the course. Nevertheless, all the remaining components will also be tested.

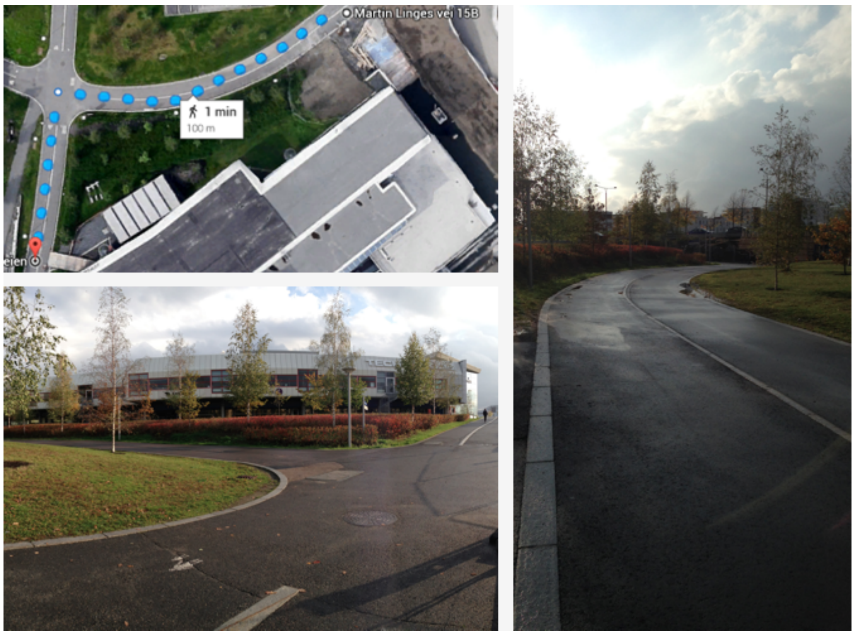


Figure 5.2: Scenario2: **right** - front view, **top left** - overview, **bottom left** - side view

As a final test course, and to make sure that the developed solution is suited for desired scenario, the **TSAG** system was also tested in the athletic tracks of *Parque de Jogos Primeiro de Maio*, in Lisbon, Portugal (coordinates $38.750931666666666N, 9.138315W$). The tests performed in the track were consisted by walking and running the official 100m race that the athletes have to perform during the olympic events.

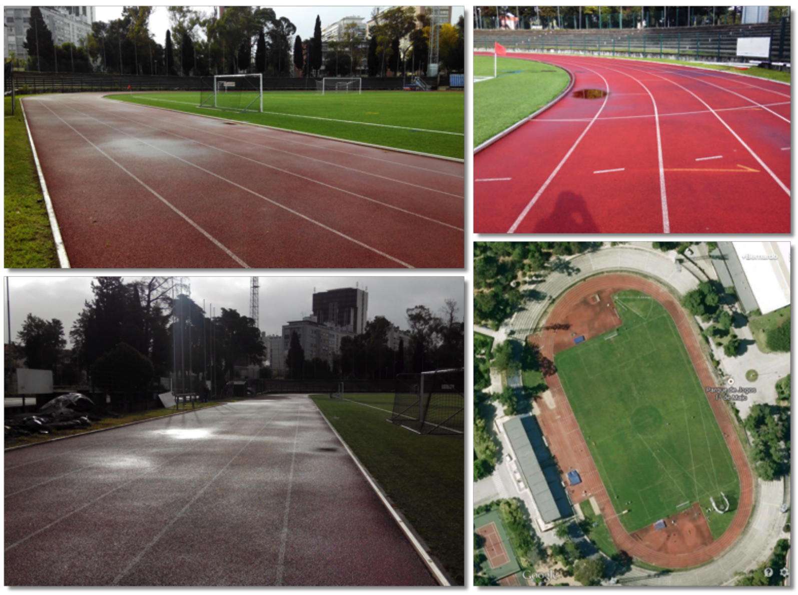


Figure 5.3: Running Track: **top left** - side view, **top right** - track curve with stripes, **bottom left** - front view, **bottom right** - over view

5.1.2 Test Device

The device that will be used for testing the **TSAG** system will be the **BQ Aquaris E5** with the following specifications that are relevant for this work - in table 5.1, with the price value of 250€. The device has the 4.4.2 version of the *Android* MOS (also known as *KitKat*) but the system/application can work in devices with earlier versions. In this device, the TTS module is already activated but with no default language and the only location provider enabled is the device's GPS built-in receiver.

Table 5.1: Test Device Specifications

CPU	MediaTek MT6592 2.0GHz True8Core
RAM Memory	2 GB
GPS receiver(s)	GPS e A-GPS
Sensors	Luminosity, Proximity, Accelerometer, Compass, Gyroscope

5.2 Results

The following results are the compilation of the output of this device's sensors and overall specification combination of the device. The results may vary if tried in another device but it won't affect the expected (and final) result.

5.2.1 Step Estimation

As presented in 5.1, the step estimation test will evaluate the accuracy of the **TSAG** system, so the following evaluation formulas will be used:

$$Step\_Error = abs(TSAG\_Step\_Count - User\_Step\_Count)$$

$$Step\_Accuracy = 1 - \frac{Step\_Error}{User\_Step\_Count} * 100\%$$

$Step\_Error$ will evaluate how many steps (above or under) the system detected with the ones the user counted while $Step\_Accuracy$ will establish the accuracy obtained in each course taken. With the evaluation formulas established, herein is presented the results of the **TSAG** regarding step estimation.

The charts available in Appendix B (from B.1 to B.4) represent samples of the accelerometer output data in each scenario and in each activity and the highest values are the ones that the system will considered as steps (highlighted as red dots).

Table 5.2: TSAG results - Step Estimation

Scenarios:	Steps Estimated*	Steps Counted	Step Error	Step Accuracy
S1_Walking	126	120	6	95%
S2_Walking	163	160	3	98,125%
S1_Running	107	110	3	97,273%
S2_Running	129	130	1	99,231%
RT_Walking	152	150	2	98,7%
RT_Running	140	137	3	97,8%

* *Steps the **TSAG** system has obtained.*

5.2.2 Distance Estimation

The distance estimation test will have the same approach as the step estimation test, in which it will consist on evaluating the accuracy of the system by comparing the distance estimation obtained from the system and the distance (from the start and finish position coordinates) that was established the 5.1 - 100m.

$$Distance\_Error = abs(TSAG\_Distance\_Estimation - 100m)$$

$$Distance\_Accuracy = 1 - \frac{Distance\_Error}{100m} * 100\%$$

$Distance\_Error$ will evaluate how much distance was (above or under) estimated by the system comparing with distance pre-defined while $Distance\_Accuracy$ will establish the accuracy obtained in each course taken. With the evaluation formulas established, herein is presented the results of the **TSAG** regarding distance estimation.

Table 5.3: TSAG results - Distance Estimation

Scenarios:	Dist. Estimated(m)*	Dist. Defined(m)	Dist. Error(m)	Dist. Accuracy
S1_Walking	88.68	100	11,32	88,7%
S2_Walking	114.02	100	14,02	85,98%
S1_Running	75.31	100	24,69	75,31%
S2_Running	90.79	100	10,79019	90,8%
RT_Walking	106.7	100	6.7	93.72%
RT_Running	84.46	100	16.46	84.46%

* *Distance the **TSAG** system has estimated.*

5.2.3 Position Estimation

The other separate test, as explained in 5.1, will evaluate the precision of the **TSAG** positioning module, but it can also be evaluated as in how accurate it was by comparing the distance the module has tracked by obtaining the distance between the first and last position in the log and comparing to the distance defined - 100 m. To evaluate the distance accuracy, we used the same formulas presented above and to evaluate the precision, the points that are further away from the original course will be considered as the maximum error factor. Since the *GPS Visualizer* uses the *Google Maps™* framework, it is possible to measure the distance between the position by both the GPS or the PDR modules and the position defined by the test course. The figures 5.4 and 5.5 are a representation of the log of both positioning modules (PDR illustrated as a red trace and GPS illustrated as a blue trace). With the evaluation formulas established, herein is presented the results of the **TSAG** regarding positioning estimation.

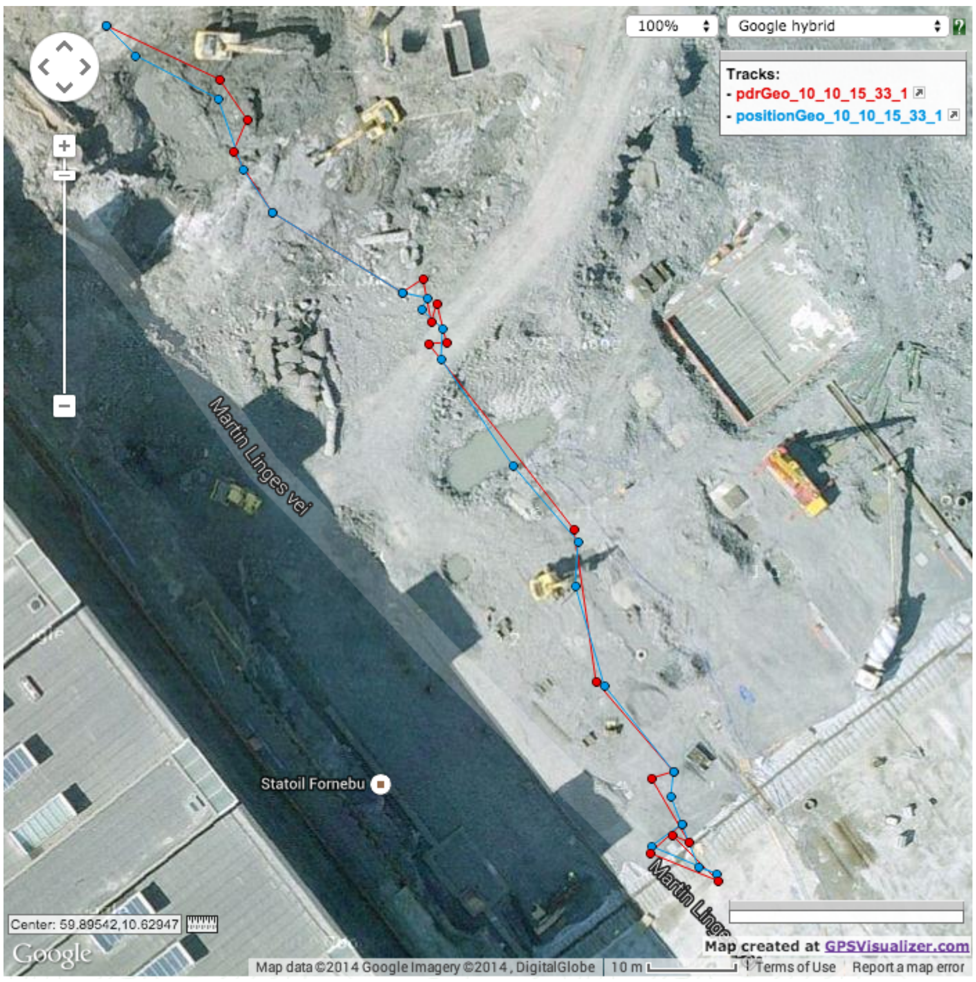

100%
Google hybrid
Tracks:
- pdrGeo_10_10_15_33_1
- positionGeo_10_10_15_33_1
Martin Linges vei
Statoil Fornebu
Martin Linges
Center: 59.89542,10.62947
Google
Map data ©2014 Google Imagery ©2014, DigitalGlobe
10 m
Terms of Use
Report a map error
Map created at GPSVisualizer.com

Figure 5.4: GPS and PDR samples in Scenario1 - Running

Table 5.4: TSAG results - Positioning (Distance Estimation - Length wise)

Scenarios:	Dist. Estimated(m)*	Dist. Defined	Dist. Error(m)	Dist. Accuracy
S1_Walking	†	100	†	†
S2_Walking	100	100	0	100%
S1_Running	≈ 82	100	≈ 18	≈ 82%
S2_Running	≈ 95	100	≈ 5	≈ 95%
RT_Walking	93,54	100	6,46	93,54%
RT_Running	110,6	100	10,6	90,41%

* *Steps the* ***TSAG*** *system has obtained.*
† *GPS Receiver Error - Insufficient Data*

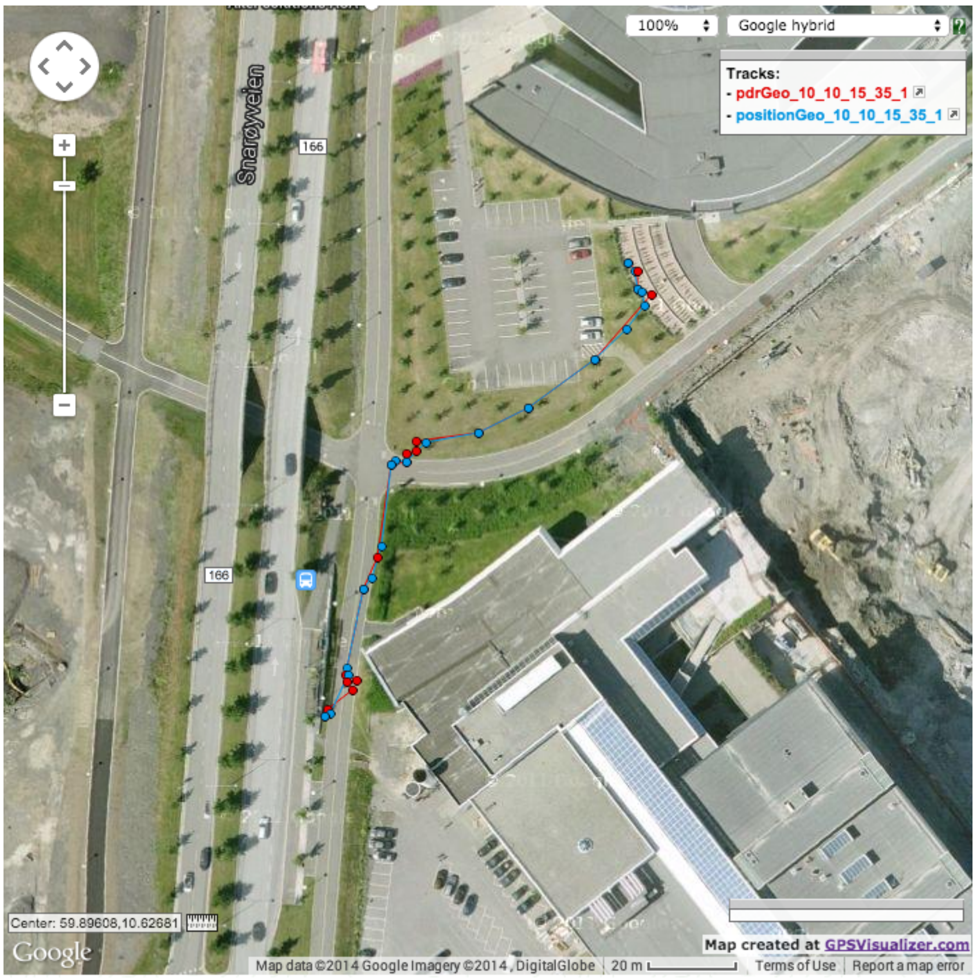

100%
Google hybrid
Tracks:
- pdrGeo_10_10_15_35_1
- positionGeo_10_10_15_35_1
Snarøyveien
166
166
Center: 59.89608,10.62681
Google
Map data ©2014 Google Imagery ©2014, DigitalGlobe
20 m
Map created at GPSVisualizer.com
Terms of Use
Report a map error

Figure 5.5: GPS and PDR samples in Scenario2 - Walking

The samples B.5 and B.6, in Appendix B, illustrate the positioning values obtaining in the running tracks.

5.2.4 Orientation and Deviation

As part of the full system test, the component that was subject to evaluation was the orientation component with the ability of detection of possible deviations by the athlete. The first scenario is a straight track, so the only possible deviations detected that can be acceptable are the ones made by the user while travelling through the course. In the second scenario however, it is expectable to detect a deviation to the left (the intersection in the middle of the course) and a right deviation at the final. The charts represent the deviations detected by the system in each course.

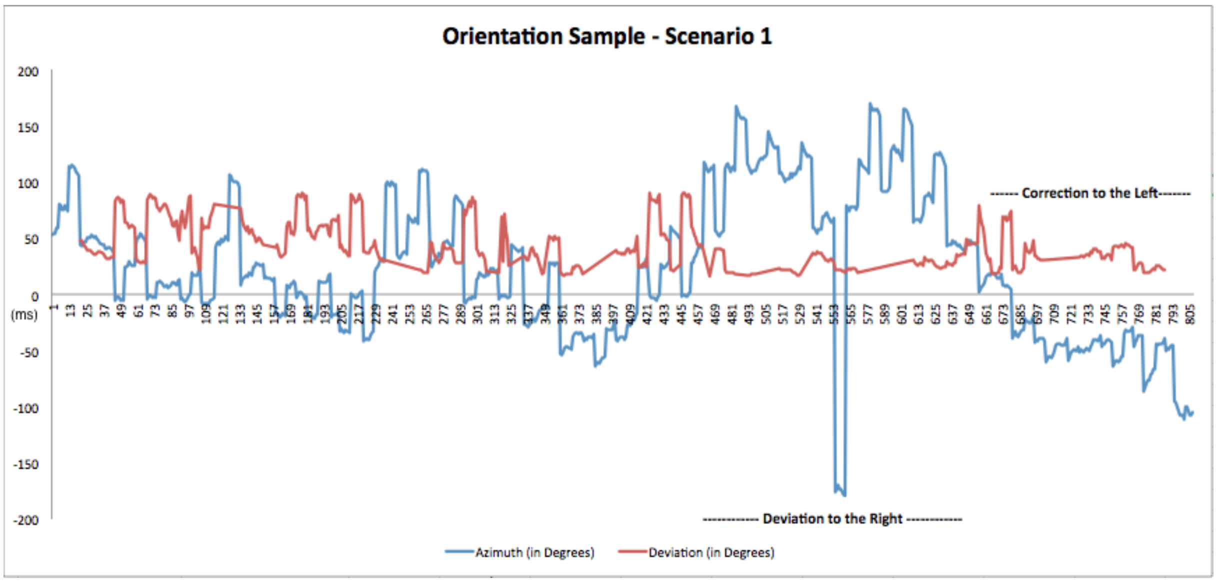


Figure 5.6: Orientation sample of Scenario1

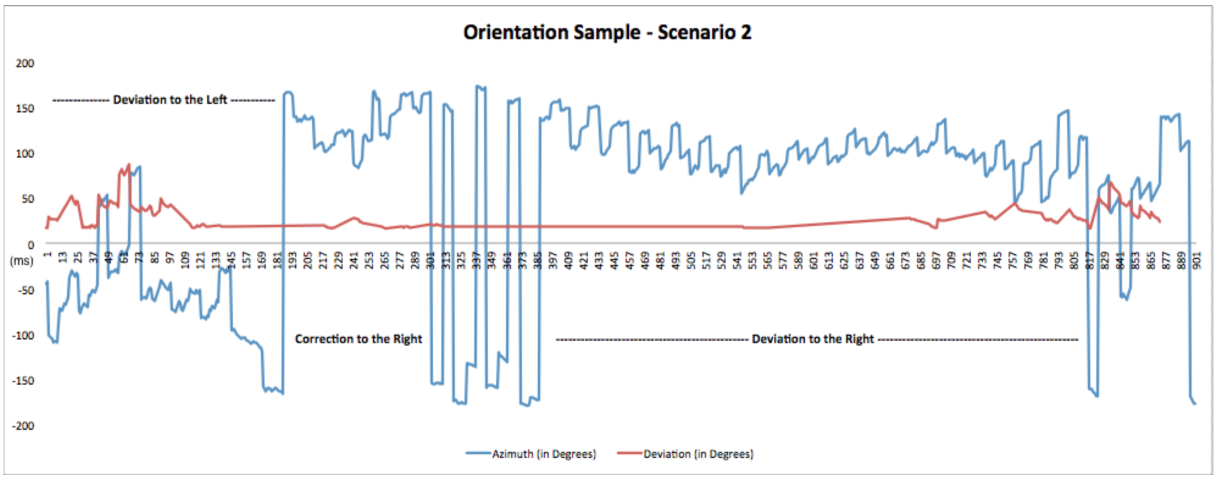


Figure 5.7: Orientation sample of Scenario2

5.3 Evaluation

After analysing the results obtained from the **TSAG** system, it is possible to conclude that:

The step estimation module obtained accuracy levels between 95% and $99,2\%$, meaning that the amount of steps obtained by this module are feasible and it is possible to consider it as fairly accurate. As shown in the charts in Appendix B (from B.1 to B.4), the difference between activities is quite noticeable by analysing the gap between peaks (a gap between peaks in a running activity is nearly half as a gap between peaks in a walking activity), but it does not influence the module behaviour. Although there were still a few steps miscounted in each activity (maximum of 6 steps miscounted in the tests), it does not undermine the overall performance of the module.

The distance estimation module obtained accuracy levels between 75% and 90%, meaning that the module needs a little improvement. The cause for these values can be related to the fact the stride factor it is not properly defined and needs further adjustment due to the fact the system needs that calibration phase in order to perform at a optimal state, hence the 75% from the first test course up to the 90% of the second test course (the system had been used and therefore calibrated at least once).

The positioning estimation module obtained accuracy levels (distance estimation from the GPS coordinates) between 82% and 100%, meaning that, accuracy wise, the results obtained by the positioning module can be considered as feasible and also accurate. Regarding the precision test, the biggest gap detected (between the defined test course and course detected by both the GPS receiver and the PDR algorithm) was around $10m$ (in width), a high value, but it is due to the fact that the surroundings of these courses had a lot of obstacles (e.g. buildings) that can difficult the retrieval of a precise location. However, in tests performed in the running track, the biggest gap detected was around $2.44m$, which is equivalent to two track lanes.

The orientation test results were able to show us the ability of the **TSAG** system to perceive rotations beyond the ones the user's body makes, alerting the user of such event. The user correction was almost immediate, as shown in the charts, allowing us to conclude that the user was able to interact and adapt correctly with the system.

Overall, the system has shown fairly good results and good performance, as is it shown above. However, there are a few improvements that can still be made as well as more tests in the desired scenario - the race track.

5.4 Summary

In this chapter, an evaluation system and methodology were presented in order to evaluate various aspects of the **TSAG** system, such as accuracy and precision.

The results obtained from the system allow us to conclude that the system has a fairly good performance with few step miscounts and low gap distances while positioning the user in the chosen track. However, the distance estimation module shows that there are some improvements that can still be made, mostly due to the stride factor. It was also shown that the interaction of the user with the system was fairly good, due to the orientation and deviation test, since the user responded almost immediately after a speech aid from the system informing of a deviation event was given.

6 Conclusion

Contents

After the **TSAG** system has been presented, by being described through its design and feature specifications and also by describing its implementation (with the usage of several algorithms developed for the *Android* environment), tests were made to the system and its results were shown and evaluated in the previous chapter. In this chapter, a recap of all the described work is presented as well as some final considerations taking in account the obtained results. Also, considerations for future work are also made.

6.1 Conclusions

Even today, visually impaired people have serious autonomy limitations performing various activities, such as in sports, due to lack of guidance systems. With this motivation, the work herein presented proposes a new tracking and guidance solution for visually impaired athletes autonomously so they can practise and compete in sporting events (e.g. running trials).

The state of the art solutions have shown promising results by providing Pedestrian Dead Reckoning (PDR) algorithms, allowing to track and log an athlete's course, knowing its current position and orientation. Other complete systems even allow the visual impaired people to navigate freely using various types of feedback such as vibratory or auditory signals. However, these systems are not adequate to the practise of sports.

To provide a low-cost solution, it was considered that, due to its specifications, a smartphone with the MOS Android (taking advantage of the wide development documentation, API capabilities and portability) was selected as the most adequate platform for the proposed system.

The proposed solution will have a calibration system which allows to learn about the athlete working on the background, while the main component, the tracking system, will track the athlete's step amount and orientation during a trial. Also, it will have a positioning component that uses the device's GPS built-in receiver as well as a PDR algorithm that is able to track the athlete's position in case of loss of GPS signal. Another feature implemented in the system is a feedback provider algorithm composed by vibratory patterns and a TTS module in order to inform the athlete of various events, but mainly it will alert the athlete in case of mispositioning of the athlete in the track lane, and it will guide him to correct the deviation, allowing it to maintain itself in the correct lane.

In order to implement such a system in the *Android* environment, various libraries or API's were used, such as *SensorManager*, which allows to access the desired sensors and retrieve their data or the *LocationManager*, which allows to obtain positioning coordinates by a certain provider (in case of this work, the GPS). Other external libraries were also used, for instance, to obtain the destination positioning coordinates by giving a start coordinate set with the distance travelled at a certain moment.

The system was subject to various steps in order to evaluate the components accuracy and precision, but also as an overall performance. User interaction and adaptability to the system was also taken into account, due to the fact that the feedback provider module must be quite efficient in order to be able to guide the athlete properly. The results obtained showed us that the general accuracy levels are between 80% to 99%, which makes it able to consider that the **TSAG** system provides fairly accurate results.

6.2 Final Remarks on Contributions

Regarding our requirements and goals defined in 1, we have made the following contributions:

- a **tracking** component that has obtained accuracy levels over 95% on step estimation and from 75% to 94% on distance estimation, which establish a slight **improvement** over the current algorithms and systems available for a **portable** environment;
- a **positioning** component that is able to provide positioning coordinates at a maximum error of $10m$ in width (surrounded by obstacles that obstruct the capture of the GPS signal), but on the running tracks, the biggest gap detected was around $2,44m$, which is equivalent to two track lanes. Also, this module obtained accuracy levels (for the distance between the first and last coordinate set obtained) over 82%, making the usage of the GPS (location provider and receiver) stable, fair and fitted for this system;
- the creation of a **feedback provider**, allied with the TTS module and vibratory patterns that allow to guide the athlete throughout the entire race without any auxiliary aid;
- Finally, the creation of a **low-cost** all-in-one solution, in which all the components and features mentioned above were developed for a **portable** device with built-in sensors and receivers.

6.3 System Limitations and Future Work

As stated in 5, some modules can be subject to further improvement as well as other approaches can be taken. Regarding the distance estimation module, the stride factor, that allows to define the distance travelled when a step was taken, can be subject to alterations in order to find a model that can work with all the physical characteristics of the athletes, mainly height and weight. Although the used factor was able to provide good results, the optimisation of this component will give a huge improvement to the module.

The developed system focus on the fact that in the running track the horizontal aid stripes (to warn the athlete that the curve is approaching) so the system does not need to know the track in advance. Other approaches could also be taken, for instance, matching the distance travelled by the athlete and matching with the established values by the **IAAF**. However, one of the purposes of this work, which is the ability (of the athlete) to race along track by its own and not with its runner guide still prevails, but the athlete will always need its trainer to train in track.

A different approach that can be considered as future work is the retrieval of positioning coordinates. Instead of using the GPS built-in receiver to obtain the constellation and afterwards retrieve the positions (a procedure that takes several minutes), it would be possible to have alongside the track a base-station/beacon system that could give reference points regarding the athlete's position automatically. The communication between the base system and **TSAG** could be through *Wi-Fi* or another reliable protocol, and the system would not take as long as it takes now to obtain the first valid position. This would allow to implement more precise positioning techniques, such as DGPS, which can obtain positioning values with the its precision in millimetres.

Bibliography

[1] U. Roentgen and G. Gelderblom, "Inventory of Electronic Mobility Aids for Persons with Visual Impairments: A Literature Review." *. . . of Visual Impairment & . . .*, no. November, 2008. [Online]. Available: http://www.eric.ed.gov/ERICWebPortal/recordDetail?accno=EJ823872

[2] S. Kammoun, G. Parseihian, O. Gutierrez, a. Brilhault, a. Serpa, M. Raynal, B. Oriola, M.-M. Macé, M. Auvray, M. Denis, S. Thorpe, P. Truillet, B. Katz, and C. Jouffrais, "Navigation and space perception assistance for the visually impaired: The NAVIG project," *Irbm*, vol. 33, no. 2, pp. 182–189, Apr. 2012. [Online]. Available: http://linkinghub.elsevier.com/retrieve/pii/S1959031812000103

[3] J. Sánchez and N. de la Torre, "Autonomous navigation through the city for the blind," *Proceedings of the 12th international ACM SIGACCESS conference on Computers and accessibility - ASSETS '10*, p. 195, 2010. [Online]. Available: http://portal.acm.org/citation.cfm?doid=1878803.1878838

[4] A. Helal, "Drishti: An integrated navigation system for visually impaired and disabled," *. . . , 2001. Proceedings. Fifth . . .*, no. 45, pp. 149–156, 2001. [Online]. Available: http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=962119

[5] J. M. Loomis, J. R. Marston, R. G. Golledge, and R. L. Klatzky, "Personal Guidance System for People with Visual Impairment: A Comparison of Spatial Displays for Route Guidance." *Journal of visual impairment & blindness*, vol. 99, no. 4, pp. 219–232, Jan. 2005. [Online]. Available: http://www.pubmedcentral.nih.gov/articlerender.fcgi?artid=2801896&tool=pmcentrez&rendertype=abstract

[6] M. Liu, "A Study of Mobile Sensing Using Smartphones," *International Journal of Distributed Sensor Networks*, vol. 2013, 2013. [Online]. Available: http://www.hindawi.com/journals/ijdsn/aip/272916/

[7] Asad-Uj-Jaman. (2011, December) Sensors in mobile phones. http://mobiledeviceinsight.com/2011/12/sensors-in-smartphones/. [Online]. Available: http://mobiledeviceinsight.com/2011/12/sensors-in-smartphones/

[8] "MEMS technical report," Yole Dévelopment, Tech. Rep., 2010.

[9] A. Godfrey, R. Conway, D. Meagher, and G. Ólaighin, “Direct measurement of human movement by accelerometry,” vol. 30, pp. 1364–1386, 2008.

[10] B.-i. Accelerometer, M. Mladenov, M. Mock, and S. Augustin, “A Step Counter Service for Java-Enabled Devices Using a Accelerometer,” pp. 1–5, 2009.

[11] A. D. Team. (2013) Sensor event: Accelerometer data. http://developer.android.com/reference/android/hardware/SensorEvent.html. [Online]. Available: http://developer.android.com/reference/android/hardware/SensorEvent.html

[12] G. Milette and A. Stroud, *Professional Android Sensor Programming*, 2012.

[13] N. Badrudino, R. Chaves, and J. Sanguino, “Sistema de Orientação Autónomo De Atletismo Para Deficientes Visuais Objectivos do Trabalho.”

[14] J. Loomis, R. Golledge, and R. Klatzky, “GPS-based navigation systems for the visually impaired.” 2001. [Online]. Available: http://psycnet.apa.org/psycinfo/2001-16260-013

[15] L. Ran, S. Helal, and S. Moore, “Drishti: an integrated indoor/outdoor blind navigation system and service,” *Second IEEE Annual Conference on Pervasive Computing and Communications, 2004. Proceedings of the*, pp. 23–30, 2004. [Online]. Available: http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=1276842

[16] N. Al-Salihi, “Precise positioning in real-time using GPS-RTK signal for visually impaired people navigation system,” no. September, 2010. [Online]. Available: http://dspace.brunel.ac.uk/handle/2438/4773

[17] J. Reis, J. Sanguino, and A. Rodrigues, “Influence of Baseline Length in Single-Frequency Real-Time Vehicle Heading Estimation,” *The 13th International Symposium on Wireless . . .*, no. Wpmc, 2010. [Online]. Available: http://scholar.google.com/scholar?hl=en&btnG=Search&q=intitle:Influence+of+baseline+length+in+single-frequency+real-time+vehicle+heading+estimation#0

[18] R. Ivanov, “Real-time GPS track simplification algorithm for outdoor navigation of visually impaired,” *Journal of Network and Computer Applications*, vol. 35, no. 5, pp. 1559–1567, Sep. 2012. [Online]. Available: http://linkinghub.elsevier.com/retrieve/pii/S1084804512000409

[19] U. Shala and A. Rodriguez, “Indoor positioning using sensor-fusion in android devices,” no. September, 2011. [Online]. Available: http://hkr.diva-portal.org/smash/record.jsf?pid=diva2:475619

[20] A. Pratama and R. Hidayat, “Smartphone-based Pedestrian Dead Reckoning as an indoor positioning system,” *System Engineering and . . .*, no. 2, 2012. [Online]. Available: http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=6339316

[21] T. Choudhury, J. Hightower, A. Rahimi, A. Rea, B. Hemingway, K. Koscher, J. A. Landay, J. Lester, and D. Wyatt, “An Embedded Activity Recognition System,” pp. 32–41, 2008.

[22] Y. Huang, H. Zheng, and C. Nugent, “Activity monitoring using an intelligent mobile phone: a validation study,” *Proceedings of the 3rd . . .*, 2010. [Online]. Available: http://dl.acm.org/citation.cfm?id=1839306

[23] A. Brajdic and R. Harle, “Walk Detection and Step Counting on Unconstrained Smartphones,” pp. 225–234, 2013.

[24] C. F. C. S. d. J. Simões, “AnDReck : Positioning Estimation using Pedestrian Dead Reckoning on Smartphones,” Master’s thesis, Instituto Superior Técnico - Taguspark, 2013.

[25] D. Pai, I. Sasi, P. S. Mantripragada, M. Malpani, and N. Aggarwal, “Padati: A Robust Pedestrian Dead Reckoning System on Smartphones,” *2012 IEEE 11th International Conference on Trust, Security and Privacy in Computing and Communications*, pp. 2000–2007, Jun. 2012. [Online]. Available: http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=6296236

[26] D. Cho, M. Mun, and U. Lee, “AutoGait: A mobile platform that accurately estimates the distance walked,” *. . . (PerCom), 2010 IEEE . . .*, 2010. [Online]. Available: http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=5466984

[27] A. Å kerberg, M. Lindén, and M. Folke, “How Accurate are Pedometer Cell Phone Applications?” *Procedia Technology*, vol. 5, no. 0, pp. 787–792, Jan. 2012. [Online]. Available: http://linkinghub.elsevier.com/retrieve/pii/S221201731200518X

[28] Nike. (2013) Nike+ running app. http://nikeplus.nike.com/plus/products/gps_app/#how_it_works_section/. [Online]. Available: http://nikeplus.nike.com/plus/products/gps_app/#how_it_works_section/

[29] Strava. (2013) Strava run app. http://www.strava.com/features. [Online]. Available: http://www.strava.com/features

[30] M. Pielot, B. Poppinga, and S. Boll, “PocketNavigator: vibro-tactile waypoint navigation for everyday mobile devices,” *. . . computer interaction with mobile devices . . .*, pp. 423–426, 2010. [Online]. Available: http://dl.acm.org/citation.cfm?id=1851696

[31] M. Lee, “Indoor Positioning System for Moving Objects on an Indoor for Blind or Visually Impaired Playing Various Sports,” *Journal of Electrical Engineering & Technology*, vol. 4, no. 1, pp. 131–134, 2009. [Online]. Available: http://www.dbpia.co.kr/Journal/ArticleDetail/1383991

[32] T. Novacheck, “The biomechanics of running.” *Gait & posture*, vol. 7, no. 1, pp. 77–95, Jan. 1998. [Online]. Available: http://www.ncbi.nlm.nih.gov/pubmed/10200378

[33] S. K. Agrawal, S. K. Banala, A. Fattah, V. Sangwan, V. Krishnamoorthy, J. P. Scholz, and W. L. Hsu, "Assessment of motion of a swing leg and gait rehabilitation with a gravity balancing exoskeleton," *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, vol. 15, no. 1, pp. 410–420, 2007.

[34] IAAF, *Track and field Facilities Manual 2008 Edition*, IAAF Track and Field Facilities Manual Editiorial Board, Ed. Monaco: Editions EGC, 2008. [Online]. Available: http://www.iaaf.org/download/download?filename=77c027b0-46b8-405d-9ffd-889fa28e3f6e.pdf&urlslug=IAAFTrackandFieldFacilitiesManual2008Edition-Chapters1-3

[35] L. Bagi. (2013) Pedometer app. https://code.google.com/p/pedometer/. [Online]. Available: https://code.google.com/p/pedometer/

[36] P. Lawitzki, "Application of Dynamic Binaural Signals in Acoustic Games," Ph.D. dissertation, Stuttgart Media University, 2012. [Online]. Available: http://www.thousand-thoughts.com/wp-content/uploads/MasterThesis_Lawitzki_2012.pdf

[37] R. Wingnity. (2014) Android gps service example. http://www.wingnity.com/blog/android-gps-location-address-using-location-manager/. [Online]. Available: http://www.wingnity.com/blog/android-gps-location-address-using-location-manager/

[38] T. Vincenty, "Direct and Inverse Solutions of Geodesics on the Ellipsoid with application of nested equations," *Survey Review*, vol. 33, pp. 88–93, 1975. [Online]. Available: papers2://publication/uuid/2C837255-3159-4377-B6F2-C5F6D94B3C26

[39] M. Gavaghan. (2014) Java geodesy library for gps vincenty's formulae. http://www.gavaghan.org/blog/free-source-code/geodesy-library-vincentys-formula-java/. [Online]. Available: http://www.gavaghan.org/blog/free-source-code/geodesy-library-vincentys-formula-java/

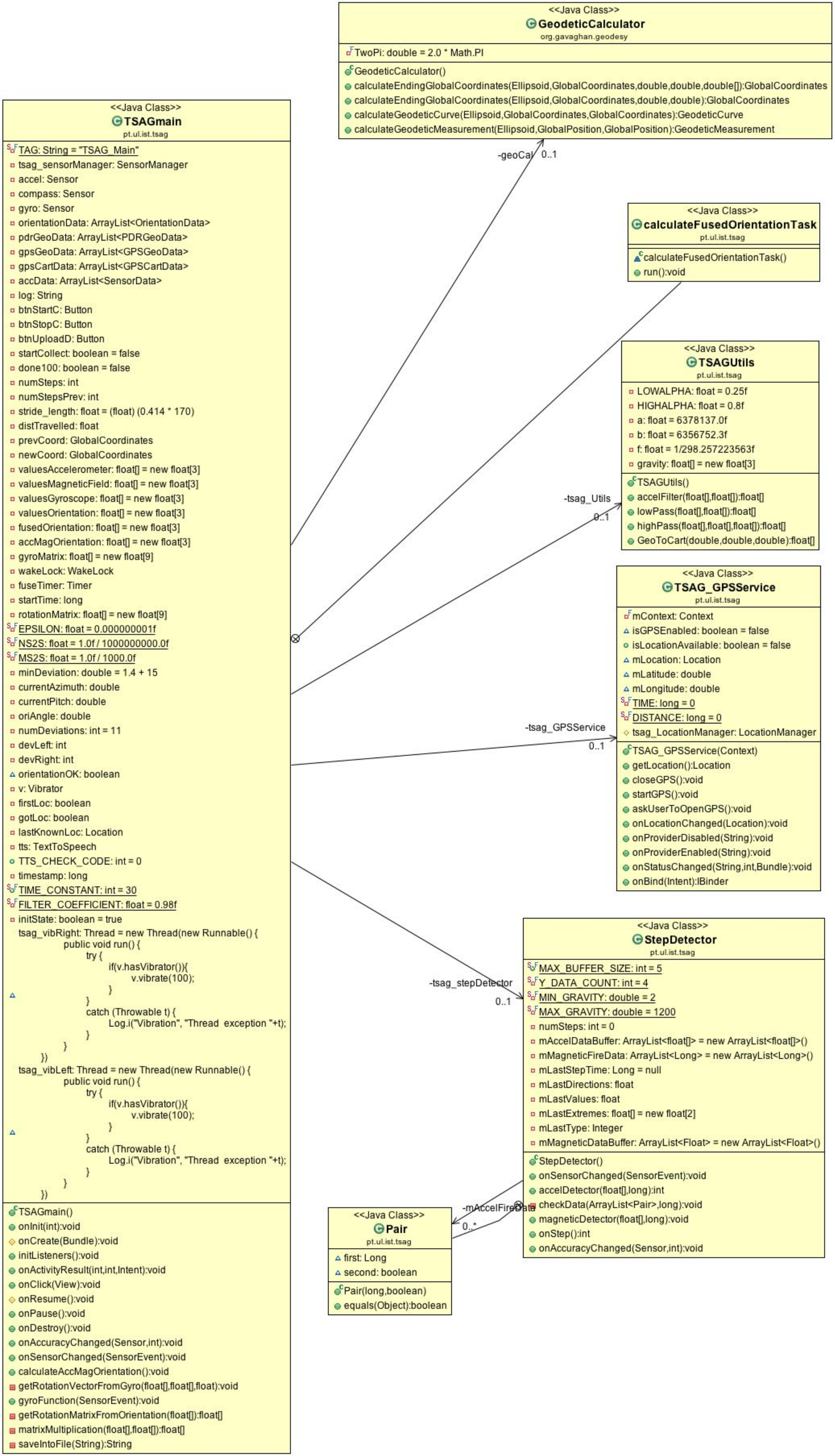
[40] I. P. Committee, *Athletics Rules and Regulations 2014-2015*, Bonn, Germany, 2014, no. January 2014.

A

Code and UML of Project

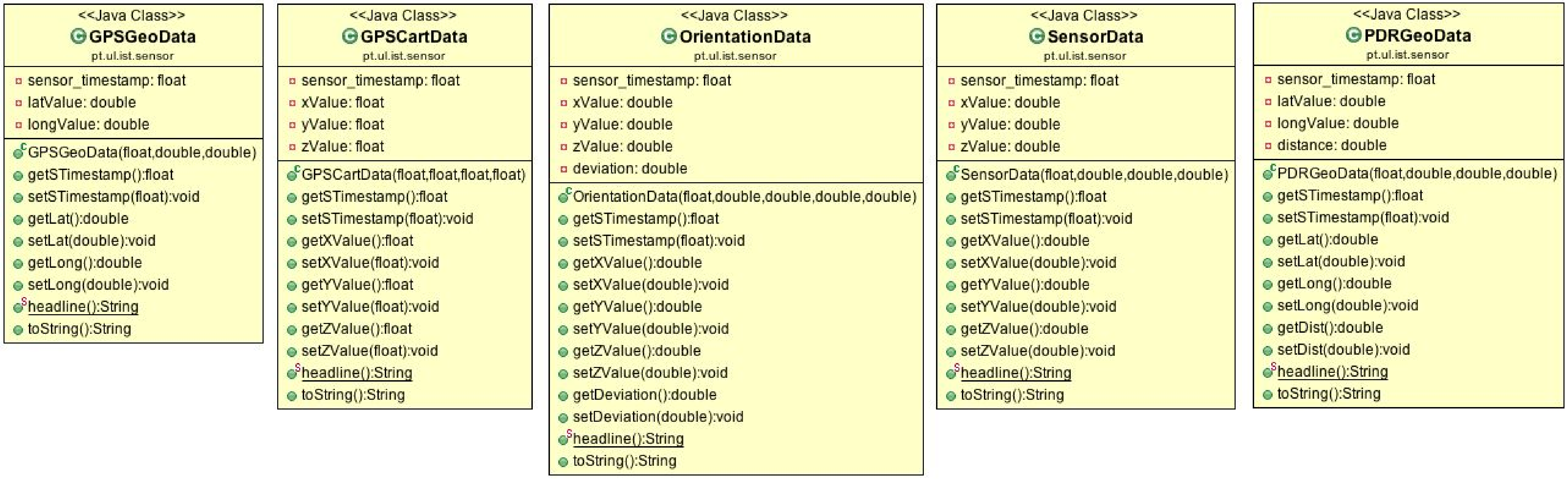
```java
            currentAzimuth = valuesOrientation[0];
            currentPitch = valuesOrientation[1];

            float normCurrentOrientation = (float) Math.sqrt(Math.pow(currentAzimuth, 2) +
                Math.pow(currentPitch, 2));
            float distBetweenOrientation = (float) Math.sqrt(Math.pow(currentAzimuth, 2));

            if(distBetweenOrientation > 0){
                oriAngle = Math.asin(distBetweenOrientation/normCurrentOrientation);
                Log.d("Deviation",Double.toString((90 - Math.toDegrees(oriAngle))));

                if((90 - Math.toDegrees(oriAngle)) > minDeviation && Math.toDegrees(
                    currentAzimuth) > 0){
                    devLeft = 0;
                    devRight++;
                    if(devRight >= numDeviations){
                        devRight = 0;
                        tts.speak((String) "Esquerda.", TextToSpeech.QUEUE_FLUSH, null);
                        Log.d("Deviation","Desvio direita! " + Math.toDegrees(oriAngle));
                        if(tsag_vibRight.isAlive()){
                            try {
                                Log.d("Vibration", "vou vibrar mais do que uma vez");
                                tsag_vibRight.join();
                            } catch (InterruptedException e) {
                                // TODO Auto-generated catch block
                                e.printStackTrace();
                            }
                        } else {
                            tsag_vibRight.run();
                            Log.d("Vibration", "vou vibrar uma vez");
                        }
                    }
                } else if ((90 - Math.toDegrees(oriAngle)) > minDeviation && Math.toDegrees
                    (currentAzimuth) < 0){
                    devRight = 0;
                    devLeft++;
                    if(devLeft >= numDeviations){
                        devLeft = 0;
                        tts.speak((String) "Direita.", TextToSpeech.QUEUE_FLUSH, null);
                        Log.d("Deviation","Desvio esquerda! " + Math.toDegrees(oriAngle));
                        if(tsag_vibLeft.isAlive()){
                            try {
                                tsag_vibLeft.join();
                            } catch (InterruptedException e) {
                                // TODO Auto-generated catch block
                                e.printStackTrace();
                            }
                        } else {
                            tsag_vibLeft.run();
                        }
                    }
                } else {
                    devRight = 0;
                    devLeft = 0;
                }
            }

            if(valuesOrientation != null){
                orientationData.add(new OrientationData(System.currentTimeMillis() -
                    startTime, (double) Math.toDegrees(valuesOrientation[0]), (double) Math
                    .toDegrees(valuesOrientation[1]),(double) Math.toDegrees(
                    valuesOrientation[2]), Math.toDegrees(oriAngle)));
            }
        }
    }
```

Listing A.1: Deviation detection algorithm



<<Java Class>>
GeodeticCalculator
org.gavaghan.geodesy
TwoPi: double = 2.0 * Math.PI
GeodeticCalculator()
calculateEndingGlobalCoordinates(Ellipsoid,GlobalCoordinates,double,double,double[]):GlobalCoordinates
calculateEndingGlobalCoordinates(Ellipsoid,GlobalCoordinates,double,double):GlobalCoordinates
calculateGeodeticCurve(Ellipsoid,GlobalCoordinates,GlobalCoordinates):GeodeticCurve
calculateGeodeticMeasurement(Ellipsoid,GlobalPosition,GlobalPosition):GeodeticMeasurement
-geoCal
0..1
<<Java Class>>
TSAGmain
pt.ul.ist.tsag
TAG: String = "TSAG_Main"
tsag_sensorManager: SensorManager
accel: Sensor
compass: Sensor
gyro: Sensor
orientationData: ArrayList<OrientationData>
pdrGeoData: ArrayList<PDRGeoData>
gpsGeoData: ArrayList<GPSGeoData>
gpsCartData: ArrayList<GPSCartData>
accData: ArrayList<SensorData>
log: String
btnStartC: Button
btnStopC: Button
btnUploadD: Button
startCollect: boolean = false
done100: boolean = false
numSteps: int
numStepsPrev: int
stride_length: float = (float) (0.414 * 170)
distTravelled: float
prevCoord: GlobalCoordinates
newCoord: GlobalCoordinates
valuesAccelerometer: float[] = new float[3]
valuesMagneticField: float[] = new float[3]
valuesGyroscope: float[] = new float[3]
valuesOrientation: float[] = new float[3]
fusedOrientation: float[] = new float[3]
accMagOrientation: float[] = new float[3]
gyroMatrix: float[] = new float[9]
wakeLock: WakeLock
fuseTimer: Timer
startTime: long
rotationMatrix: float[] = new float[9]
EPSILON: float = 0.000000001f
NS2S: float = 1.0f / 1000000000.0f
MS2S: float = 1.0f / 1000.0f
minDeviation: double = 1.4 + 15
currentAzimuth: double
currentPitch: double
oriAngle: double
numDeviations: int = 11
devLeft: int
devRight: int
orientationOK: boolean
v: Vibrator
firstLoc: boolean
gotLoc: boolean
lastKnownLoc: Location
tts: TextToSpeech
TTS_CHECK_CODE: int = 0
timestamp: long
TIME_CONSTANT: int = 30
FILTER_COEFFICIENT: float = 0.98f
initState: boolean = true
TSAGmain()
onInit(int):void
onCreate(Bundle):void
initListeners():void
onActivityResult(int,int,Intent):void
onClick(View):void
onResume():void
onPause():void
onDestroy():void
onAccuracyChanged(Sensor,int):void
onSensorChanged(SensorEvent):void
calculateAccMagOrientation():void
getRotationVectorFromGyro(float[],float[],float):void
gyroFunction(SensorEvent):void
getRotationMatrixFromOrientation(float[]):float[]
matrixMultiplication(float[],float[]):float[]
saveIntoFile(String):String
<<Java Class>>
calculateFusedOrientationTask
pt.ul.ist.tsag
calculateFusedOrientationTask()
run():void
<<Java Class>>
TSAGUtils
pt.ul.ist.tsag
LOWALPHA: float = 0.25f
HIGHALPHA: float = 0.8f
a: float = 6378137.0f
b: float = 6356752.3f
f: float = 1/298.257223563f
gravity: float[] = new float[3]
TSAGUtils()
accelFilter(float[],float[]):float[]
lowPass(float[],float[]):float[]
highPass(float[],float[],float[]):float[]
GeoToCart(double,double,double):float[]
-tsag_Utils
0..1
<<Java Class>>
TSAG_GPSService
pt.ul.ist.tsag
mContext: Context
isGPSEnabled: boolean = false
isLocationAvailable: boolean = false
mLocation: Location
mLatitude: double
mLongitude: double
TIME: long = 0
DISTANCE: long = 0
tsag_LocationManager: LocationManager
TSAG_GPSService(Context)
getLocation():Location
closeGPS():void
startGPS():void
askUserToOpenGPS():void
onLocationChanged(Location):void
onProviderDisabled(String):void
onProviderEnabled(String):void
onStatusChanged(String,int,Bundle):void
onBind(Intent):IBinder
-tsag_GPSService
0..1
-tsag_stepDetector
0..1
<<Java Class>>
StepDetector
pt.ul.ist.tsag
MAX_BUFFER_SIZE: int = 5
Y_DATA_COUNT: int = 4
MIN_GRAVITY: double = 2
MAX_GRAVITY: double = 1200
numSteps: int = 0
mAccelDataBuffer: ArrayList<float[]> = new ArrayList<float[]>()
mMagneticFireData: ArrayList<Long> = new ArrayList<Long>()
mLastStepTime: Long = null
mLastDirections: float
mLastValues: float
mLastExtremes: float[] = new float[2]
mLastType: Integer
mMagneticDataBuffer: ArrayList<Float> = new ArrayList<Float>()
StepDetector()
onSensorChanged(SensorEvent):void
accelDetector(float[],long):int
checkData(ArrayList<Pair>,long):void
magneticDetector(float[],long):void
onStep():int
onAccuracyChanged(Sensor,int):void
-mAccelFireData
0..*
<<Java Class>>
Pair
pt.ul.ist.tsag
first: Long
second: boolean
Pair(long,boolean)
equals(Object):boolean

```
tsag_vibRight: Thread = new Thread(new Runnable() {
        public void run() {
            try {
                if(v.hasVibrator()){
                    v.vibrate(100);
                }
            }
            catch (Throwable t) {
                Log.i("Vibration", "Thread  exception "+t);
            }
        }
    })
tsag_vibLeft: Thread = new Thread(new Runnable() {
        public void run() {
            try {
                if(v.hasVibrator()){
                    v.vibrate(100);
                }
            }
            catch (Throwable t) {
                Log.i("Vibration", "Thread  exception "+t);
            }
        }
    })
```

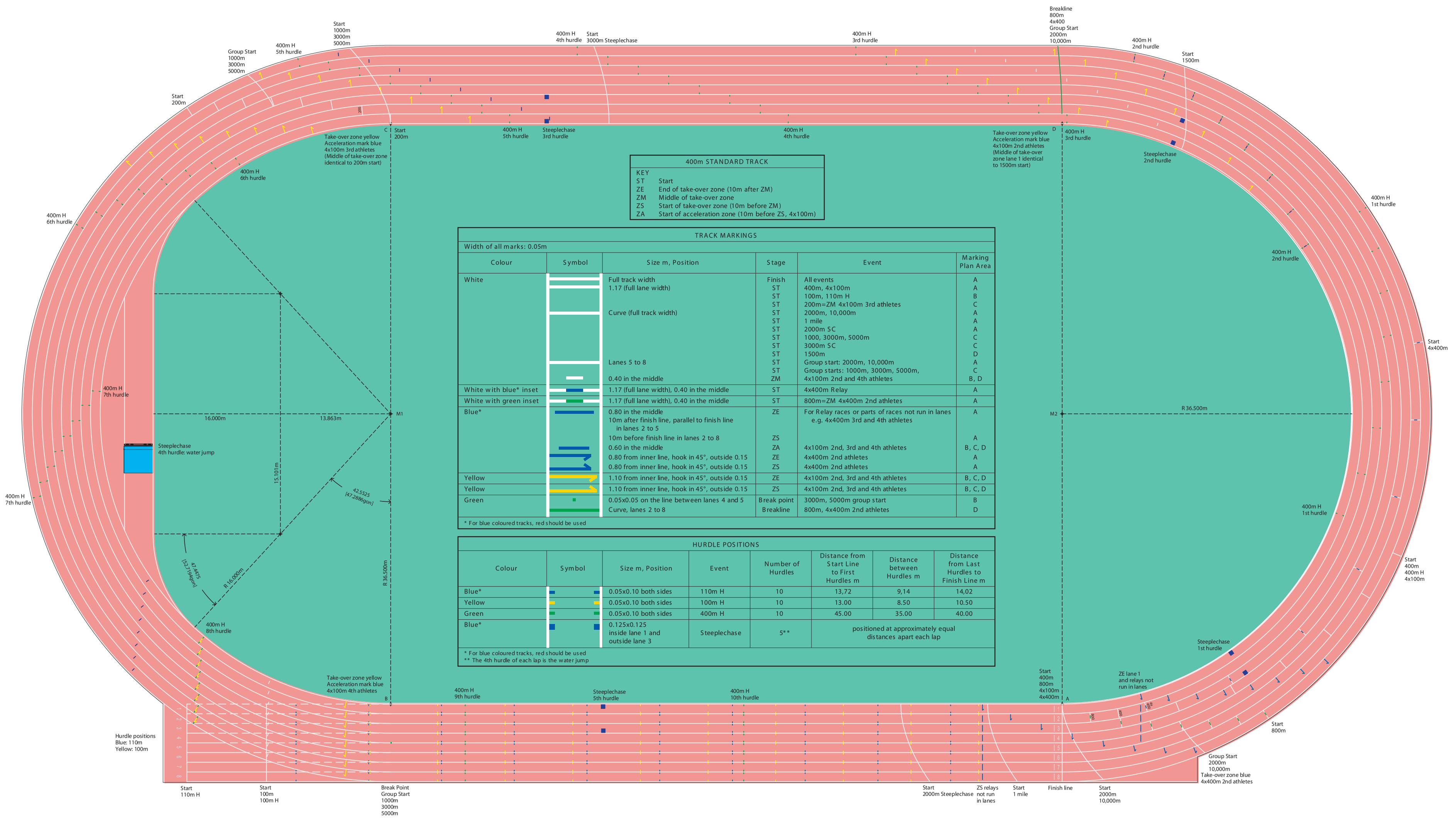

<<Java Class>>
GPSGeoData
pt.ul.ist.sensor
sensor_timestamp: float
latValue: double
longValue: double
GPSGeoData(float,double,double)
getSTimestamp():float
setSTimestamp(float):void
getLat():double
setLat(double):void
getLong():double
setLong(double):void
headline():String
toString():String
<<Java Class>>
GPSCartData
pt.ul.ist.sensor
sensor_timestamp: float
xValue: float
yValue: float
zValue: float
GPSCartData(float,float,float,float)
getSTimestamp():float
setSTimestamp(float):void
getXValue():float
setXValue(float):void
getYValue():float
setYValue(float):void
getZValue():float
setZValue(float):void
headline():String
toString():String
<<Java Class>>
OrientationData
pt.ul.ist.sensor
sensor_timestamp: float
xValue: double
yValue: double
zValue: double
deviation: double
OrientationData(float,double,double,double,double)
getSTimestamp():float
setSTimestamp(float):void
getXValue():double
setXValue(double):void
getYValue():double
setYValue(double):void
getZValue():double
setZValue(double):void
getDeviation():double
setDeviation(double):void
headline():String
toString():String
<<Java Class>>
SensorData
pt.ul.ist.sensor
sensor_timestamp: float
xValue: double
yValue: double
zValue: double
SensorData(float,double,double,double)
getSTimestamp():float
setSTimestamp(float):void
getXValue():double
setXValue(double):void
getYValue():double
setYValue(double):void
getZValue():double
setZValue(double):void
headline():String
toString():String
<<Java Class>>
PDRGeoData
pt.ul.ist.sensor
sensor_timestamp: float
latValue: double
longValue: double
distance: double
PDRGeoData(float,double,double,double)
getSTimestamp():float
setSTimestamp(float):void
getLat():double
setLat(double):void
getLong():double
setLong(double):void
getDist():double
setDist(double):void
headline():String
toString():String

B

Images

IAAF 400 METRE STANDARD TRACK, MARKING PLAN

SCALE - 1:350



400m STANDARD TRACK

KEY	
ST	Start
ZE	End of take-over zone (10m after ZM)
ZM	Middle of take-over zone
ZS	Start of take-over zone (10m before ZM)
ZA	Start of acceleration zone (10m before ZS, 4x100m)

TRACK MARKINGS

Width of all marks: 0.05m

Colour	Symbol	Size m, Position	Stage	Event	Marking Plan Area
White		Full track width	Finish	All events	A
		1.17 (full lane width)	ST	400m, 4x100m	A
			ST	100m, 110m H	B
			ST	200m=ZM 4x100m 3rd athletes	C
		Curve (full track width)	ST	2000m, 10,000m	A
			ST	1 mile	A
			ST	2000m SC	A
			ST	1000, 3000m, 5000m	C
			ST	3000m SC	C
			ST	1500m	D
		Lanes 5 to 8	ST	Group start: 2000m, 10,000m	A
			ST	Group starts: 1000m, 3000m, 5000m,	C
		0.40 in the middle	ZM	4x100m 2nd and 4th athletes	B, D
White with blue* inset		1.17 (full lane width), 0.40 in the middle	ST	4x400m Relay	A
White with green inset		1.17 (full lane width), 0.40 in the middle	ST	800m=ZM 4x400m 2nd athletes	A
Blue*		0.80 in the middle 10m after finish line, parallel to finish line in lanes 2 to 5	ZE	For Relay races or parts of races not run in lanes e.g. 4x400m 3rd and 4th athletes	A
		10m before finish line in lanes 2 to 8	ZS		A
		0.60 in the middle	ZA	4x100m 2nd, 3rd and 4th athletes	B, C, D
		0.80 from inner line, hook in 45°, outside 0.15	ZE	4x400m 2nd athletes	A
		0.80 from inner line, hook in 45°, outside 0.15	ZS	4x400m 2nd athletes	A
Yellow		1.10 from inner line, hook in 45°, outside 0.15	ZE	4x100m 2nd, 3rd and 4th athletes	B, C, D
Yellow		1.10 from inner line, hook in 45°, outside 0.15	ZS	4x100m 2nd, 3rd and 4th athletes	B, C, D
Green		0.05x0.05 on the line between lanes 4 and 5	Break point	3000m, 5000m group start	B
		Curve, lanes 2 to 8	Breakline	800m, 4x400m 2nd athletes	D

* For blue coloured tracks, red should be used

HURDLE POSITIONS

Colour	Symbol	Size m, Position	Event	Number of Hurdles	Distance from Start Line to First Hurdles m	Distance between Hurdles m	Distance from Last Hurdles to Finish Line m
Blue*		0.05x0.10 both sides	110m H	10	13,72	9,14	14,02
Yellow		0.05x0.10 both sides	100m H	10	13.00	8.50	10.50
Green		0.05x0.10 both sides	400m H	10	45.00	35.00	40.00
Blue*		0.125x0.125 inside lane 1 and outside lane 3	Steeplechase	5**	positioned at approximately equal distances apart each lap		

* For blue coloured tracks, red should be used
** The 4th hurdle of each lap is the water jump

CONSTRUCTION MEASUREMENTS	m
Construction radius of curve (including the raised kerb on inside of track)	36.500
Radius of measurement line (line of running) in lane 1 (0.30m outside raised kerb)	36.800
Length of each straight section	84.390
Legth of each bend on construction line (kerb line)	114.668
Length of each bend along line of running	115.611
Length of track on construction line (kerb line)	398.116
Length of track along line of running	400.001
Width of lanes (including 0.05m on outside)	1.220
Length of Steeplechase lap along line of running where the water jump is inside the 400m track	396.084
With the exception of lane 1, all lanes are measured 0.20m out from the outer edge of the inner line.	
All race distances are measured in a clockwise direction from the edge of the finish line nearer to the start to the edge of the appropriate line farther from the finish.	
Marking of start, relay and hurdle positions: With measuring tape on straights only; with theodolite on the bends according to the centre angles of the nominal arc segments.	
Marking with measuring tape on bends only as a backup method: E.g. checking, correcting and supplementing. In each lane, always measure from the start (A,C) or end (B,D) of the arc.	

Lane staggers in m, measurement line distance 0.20m from lane line (Width of lanes 1.22)

Distance on Line of Running	Marking Plan Area	Bends Run in Lanes	Lane 2	Lane 3	Lane 4	Lane 5	Lane 6	Lane 7	Lane 8
200	C	1	3.519	7.352	11.185	15.017	18.850	22.683	26.516
400	A	2	7.038	14.704	22.370	30.034	37.700	45.366	53.032
800	A	1	3.526	7.384	11.260	15.151	19.061	22.989	26.933
4x400	A	3	10.564	22.088	33.630	45.185	56.761	68.355	79.965

Figure 2.2.1.6a - Marking plan for the IAAF 400m Standard Track

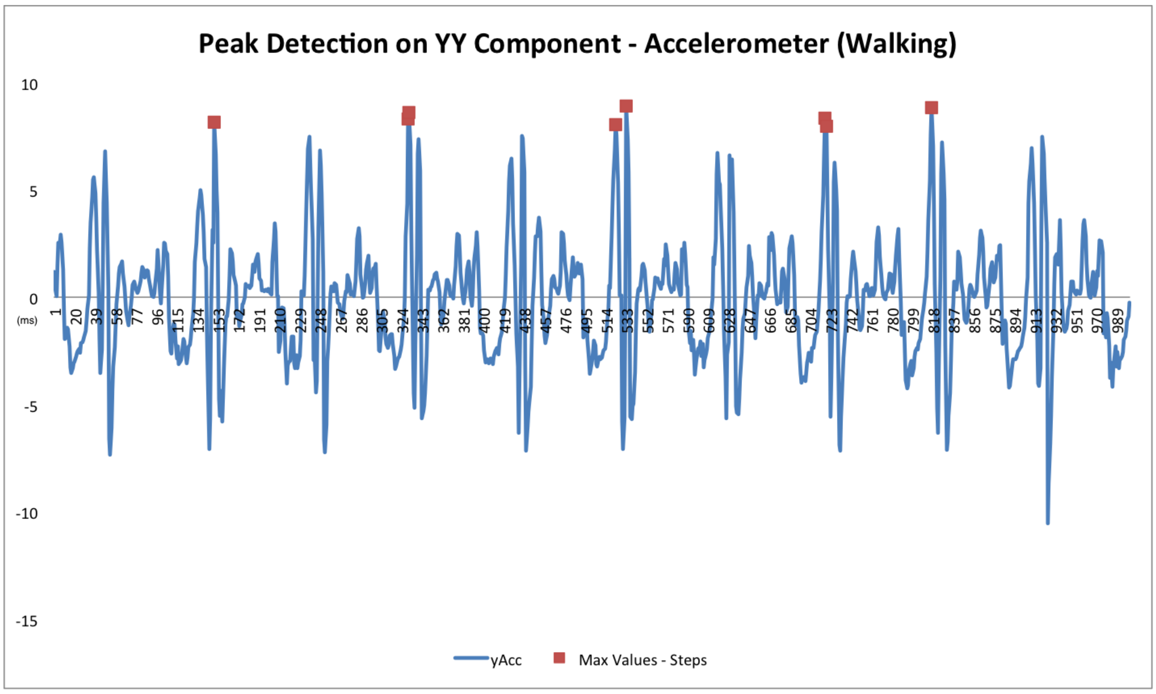


Figure B.1: Sample of Peak Detection on Scenario1 - Walking

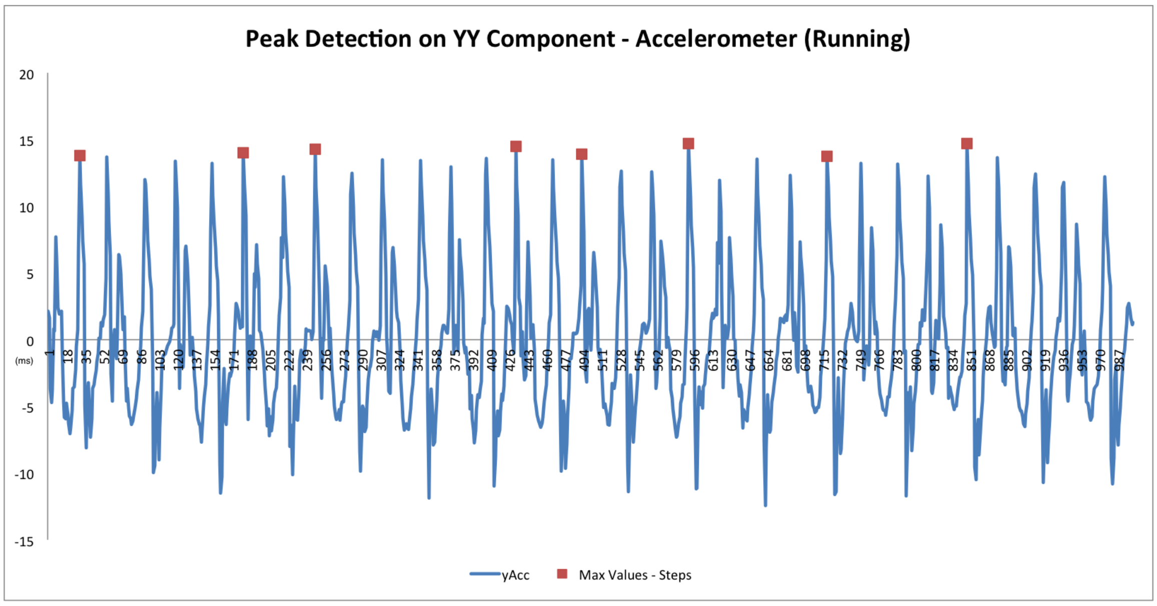


Figure B.2: Sample of Peak Detection on Scenario1 - Running

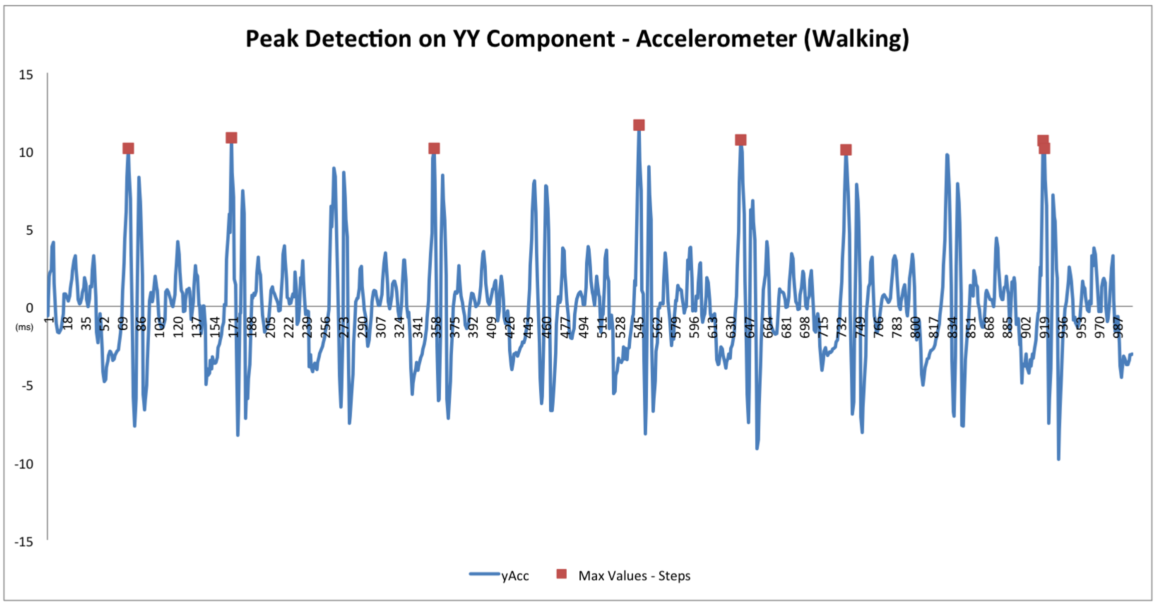


Figure B.3: Sample of Peak Detection on Scenario2 - Walking

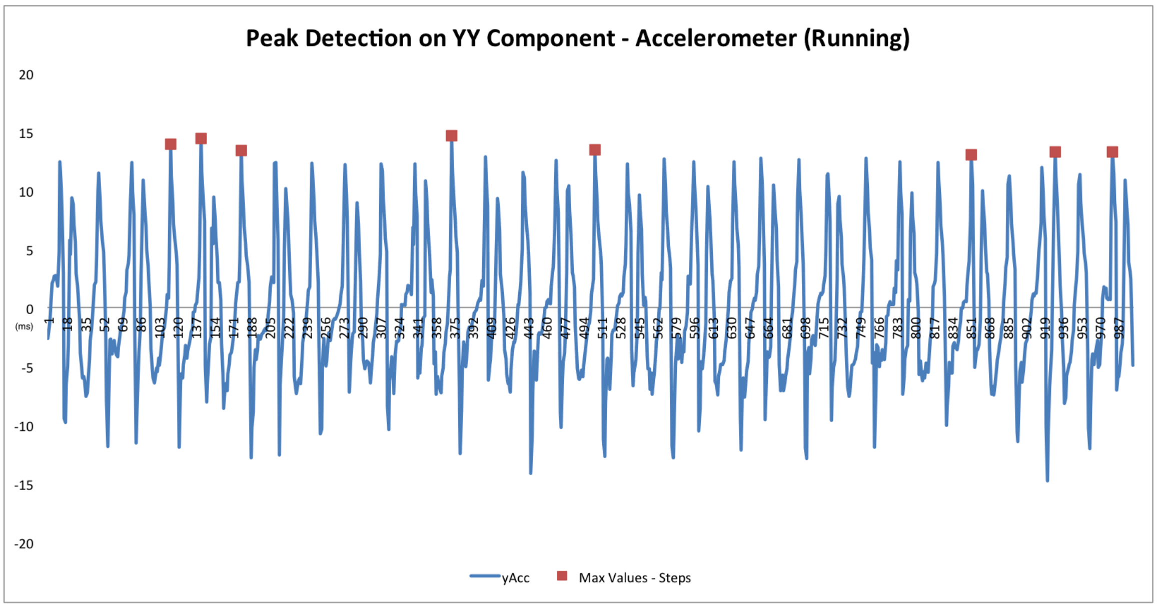


Figure B.4: Sample of Peak Detection on Scenario2 - Running

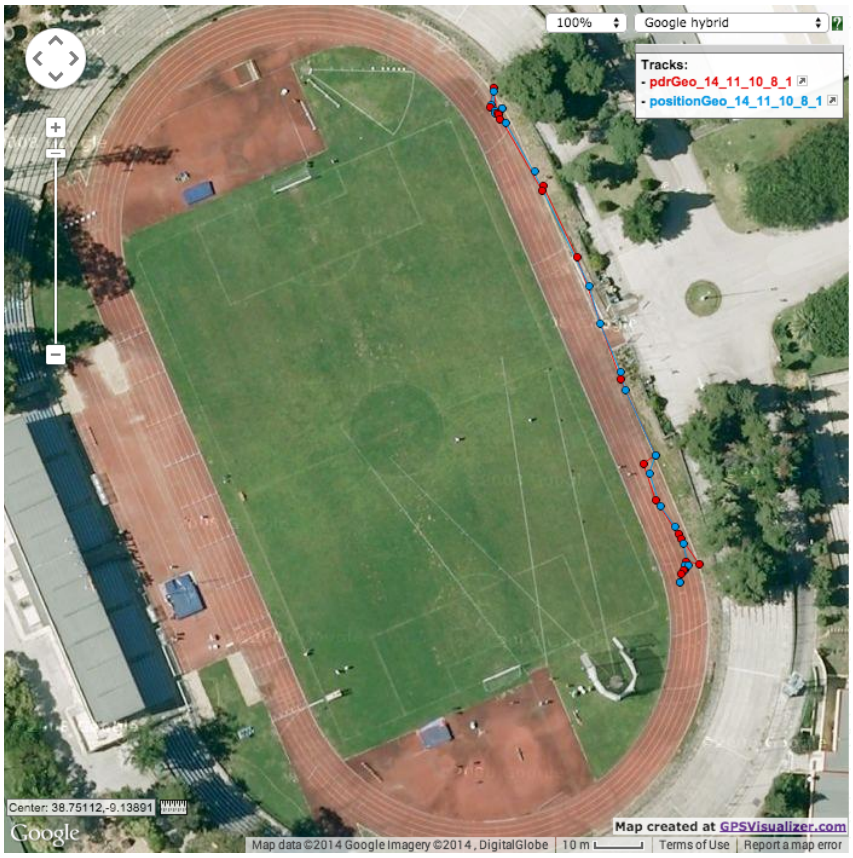

100%
Google hybrid
Tracks:
- pdrGeo_14_11_10_8_1
- positionGeo_14_11_10_8_1
Center: 38.75112,-9.13891
Google
Map created at GPSVisualizer.com
Map data ©2014 Google Imagery ©2014 , DigitalGlobe
10 m
Terms of Use
Report a map error

Figure B.5: GPS and PDR samples in Running Track - 100m

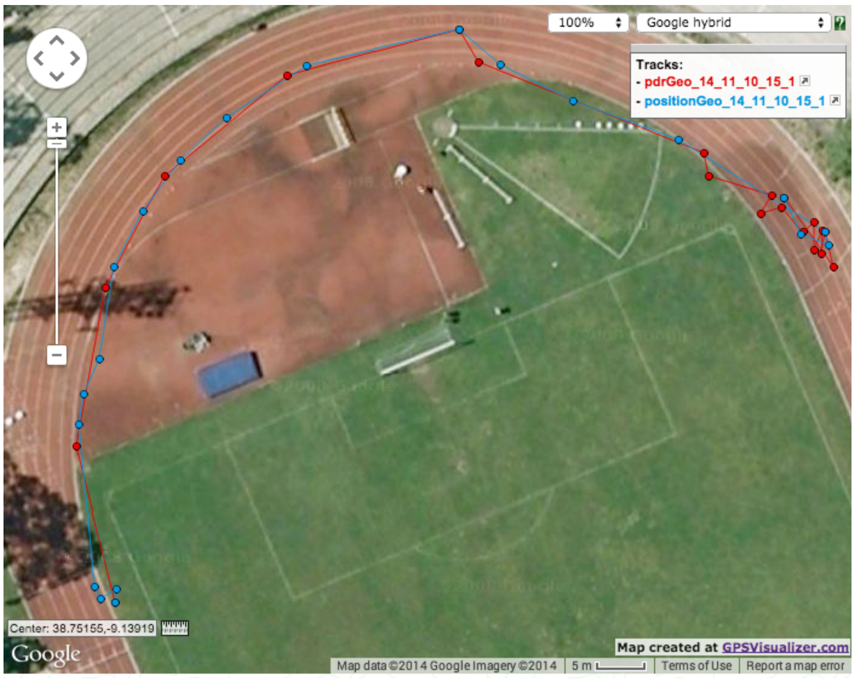

100%
Google hybrid
Tracks:
- pdrGeo_14_11_10_15_1
- positionGeo_14_11_10_15_1
Center: 38.75155,-9.13919
Google
Map created at GPSVisualizer.com
Map data ©2014 Google Imagery ©2014
5 m
Terms of Use
Report a map error

Figure B.6: GPS and PDR samples in Running Track - Curve